Deep Triplet Networks with Attention for Sensor-based Human Activity Recognition

Bulat Khaertdinov, Esam Ghaleb, and Stylianos Asteriadis
Maastricht University, Maastricht, the Netherlands
{b.khaertdinov, esam.ghaleb, stelios.asteriadis}@maastrichtuniversity.nl

***Abstract*—One of the most significant challenges in Human Activity Recognition using wearable devices is inter-class similarities and subject heterogeneity. These problems lead to the difficulties in constructing robust feature representations that might negatively affect the quality of recognition. This study, for the first time, applies deep triplet networks with various triplet loss functions and mining methods to the Human Activity Recognition task. Moreover, we introduce a novel method for constructing hard triplets by exploiting similarities between subjects performing the same activities using the concept of Hierarchical Triplet Loss. Our deep triplet models are based on the recent state-of-the-art LSTM networks with two attention mechanisms. The extensive experiments conducted in this paper identify important hyperparameters and settings for training deep metric learning models on widely-used open-source Human Activity Recognition datasets. The comparison of the proposed models against the recent benchmark models shows that deep metric learning approach has the potential to improve the quality of recognition. Specifically, at least one of the implemented triplet networks shows the state-of-the-art results for each dataset used in this study, namely PAMAP2, USC-HAD and MHEALTH. Another positive effect of applying deep triplet networks and especially the proposed sampling algorithm is that feature representations are less affected by inter-class similarities and subject heterogeneity issues.**

I. Introduction

Human Activity Recognition (HAR) is one of the classic problems in ubiquitous computing, human-computer interaction, and ambient assisted living. Modern Activity Recognition algorithms are used in various application areas which include but are not limited to health monitoring [1], smart homes [2] and manufacturing automation [3]. There are three main approaches widely used to recognize human activities, namely video-based, sensor-based and their combination. While video-based HAR methods aim to identify activities given video and/or depth data, sensor-based HAR techniques exploit time-series data obtained through wearable devices and ambient sensors.

Several issues arise when HAR is based on videos. First, privacy concerns are something which should be faced while recording video data and installing cameras in personal environments [4]. Second, a visual scene might contain other subjects. Thus, an algorithm should also be able to identify the relevant person. Another problem is that a real-time video-based HAR system might only be developed in a pre-defined scene with cameras whereas sensor-based HAR systems can be exploited everywhere when a person wears required devices. Despite the fact that issues described above are not related to sensor-based HAR, there are several aspects which make it a complex task. One of the problems is user heterogeneity, in other words, different people may perform the same activities in a different way which often introduces challenges in constructing models able to generalize [4]. Another challenge is to make use of models to relying on features able to handle inter-class similarities. Together, these issues constitute feature extraction a challenging and crucial part of building a robust HAR model.

Most recent works on sensor-based HAR utilize deep neural networks. Architectures which are typically used for this problem include Convolutional Neural Networks (CNNs) [5], [6], Recurrent Neural Networks (RNNs) [7], [8] or hybrid models containing both architectures [9], [10]. In some works, modern frameworks such as multimodal fusion ([11], [12]) and attention mechanisms [13] are adapted to sensor-based HAR. In the vast majority of studies, models are trained using a traditional end-to-end deep learning approach with a softmax output classification layer.

Deep Metric Learning (DML), also known as similarity learning, is a paradigm of learning deep feature embeddings which are extensively used in various problems, mostly coming from the Computer Vision domain. The idea of the paradigm is to group examples of the same class in a manifold while pushing examples of different classes apart from each other [14]. This approach requires specific loss functions which are based on distances between certain data points such as triplet loss [15], quadruplet loss [16] or contrastive loss [17]. In this paper, we are focused on the triplet loss function and its variations.

This study aims to apply the DML concept to sensor-based HAR. The main motivation for exploiting DML is its powerful property of extracting robust deep feature embeddings. Therefore, DML is a potential solution to address challenging problems in sensor-based HAR, namely inter-class similarities and subject heterogeneity. Specifically, we introduce a novel method for sensor-based HAR which is based on a concept of a hierarchical triplet loss function [14]. In this method, we explore how data points corresponding to activities performed by different subjects are located in a feature space and later use distances between these examples to mine informative triplets. In such a way, we force examples of the same activities to be closer to each other even though they were performed by different subjects with different movement patterns. The main contributions of our study are listed as follows:

- We apply the paradigm of DML to the sensor-based HAR task and provide extensive evaluations for three open-source datasets. The DML models used in this study are based on the LSTM networks with two attention mechanisms.
- We examine the effect of various triplet loss functions and batch sampling techniques on sensor-based HAR models. This includes a comparison of performance metrics as well as visual evaluation of a manifold using the t-SNE algorithm [18].
- We address the subject heterogeneity and feature extraction problems by designing a novel subject-based triplet mining technique to train models with hierarchical triplet loss and reduce the variability of activity clusters in a manifold space.

II. Related Work

Sensor-based Human Activity Recognition (HAR). The task of sensor-based HAR might be considered as a time-series classification problem where data is obtained through different types of devices including Inertial Measurement Units, devices for health monitoring measurements (e.g. heartbeat rate) and binary sensors. In most of the studies, long-term sensor data is segmented into short time windows [7], [9], [13]. In some works, Fast Fourier Transform is applied to the obtained short time-windows to get the spectrograms of the signals and treat them as inputs of the classification algorithms [19], [11].

Recent studies in sensor-based HAR are based on Deep Neural Networks of different CNN and RNN architectures. In 2015, Yang et al. [5] applied CNNs with the additional algorithm for sensor-unification to the existing datasets. This work was later followed by studies of Ordonez [9] and Hammerla [7], who contributed to the problem by applying a combination of CNN and LSTM and different LSTM architectures, respectively. Concerning the latest literature, many works make use of Attention Mechanisms [13], [20] and Multimodal Fusion [11], [12] while dealing with sensor-based HAR.

Feature Extraction and Subject Heterogeneity. Feature extraction and subject heterogeneity are challenging problems in sensor-based HAR [4]. The main reason for the feature extraction complexity is inter-activity similarity [21]. In other words, signals obtained from sensors might be similar for some activities (e.g. walking and Nordic walking) which have common movement patterns. Subject heterogeneity is another issue related to feature extraction which arises because samples of the same activity performed by two people might differ significantly, thus, making activity recognition a highly person-dependent problem.

The methods used to address these problems exploit various architectures of neural networks and approaches. Rokni et al. [22] used transfer learning by training CNNs on training subjects data, freezing part of the network and using a small amount of the test data to fine-tune the top layers. In [23], Generative Adversarial Networks (GANs) were used to augment a training set with synthesized data points which are similar to examples corresponding to subjects from a test set. Finally, Chen et al. [24] used several training protocols to train an autoencoder model in a self-supervised manner and preserve task-specific consistency and decrease person-specific discrepancy.

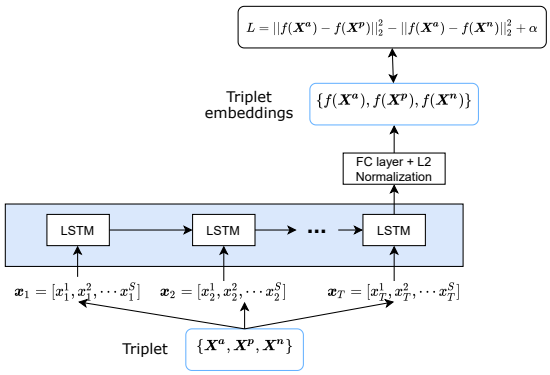


Fig. 1: The proposed baseline.

Deep Metric Learning and Triplet Loss. In recent years, deep metric learning has been extensively used in Computer Vision applications such as face recognition [15], image retrieval [25] and person re-identification [26]. The main idea is to learn models that extract robust feature representations in a manifold space instead of learning class labels in an end-to-end manner. Loss functions in deep metric learning methods are dependant on the distance between data points grouped into pairs, triplets or quadruplets. Moreover, they are designed in such a way that distances between data points of the same class are minimized while distances between different classes are maximized.

In this paper, we make an emphasis on networks which are trained using triplet loss. In this approach, data points are grouped into triplets. In each triplet, there are two examples of a positive class, so-called anchor and positive examples, and one example of another class, a negative example. The idea is to minimize the distance between the anchor and positive data points while maximizing the distance between the anchor and negative ones in an embedding space.

The main challenge which is typically addressed in recent studies on deep metric learning with triplet loss is informative triplet mining. At some point deep metric learning models can only be improved using informative and hard samples [15]. Furthermore, after several epochs, even hard triplets in a mini-batch might not significantly contribute to the gradients of learnable parameters because they do not consider the global structure of data [14]. Informative sample selection was addressed by a variety of different sampling techniques. In 2015, Schroff et al. [15] suggested mining hard triplets for which distances between anchor and positive examples are larger than distances between anchor and negative ones. This approach was later criticized in [26] because too many outliers might be considered while only the hardest triplets are selected. They also suggested using semi-hard samples which exclude the easy triplets for which the difference between the positive and negative distances is larger than a violate margin.

The methods described above are based on distances between certain data points and they do not consider the global structure of embeddings. In 2018, Ge et al. [14] proposed an approach, namely the Hierarchical Triplet Loss, that mines triplets based on the class similarities obtained from embeddings. After each epoch, they construct a hierarchical tree based on interclass distances. The tree is later exploited to perform anchor-neighbor sampling and generate informative triplets which contribute to the gradients of learnable parameters. Additionally, they introduced a novel dynamic violate margin calculated using based on the hierarchical tree.

There are a few studies which exploit deep metric learning approaches for the sensor-based HAR. For example, in [27], Mubarak et al. applied CNNs in combination with the original triplet loss. Also, Sheng et al. [28] used Siamese networks to exploit CNN-LSTM combination as a backbone model (encoder) [28]. However, these studies do not provide extensive evaluations on existing sensor-based HAR benchmarks. Besides, no studies thoroughly investigate different triplet mining mechanisms and the latest advances of deep triplet networks (such as Hierarchical Triplet Loss [14]) for the sensor-based HAR problem. Moreover, in this paper, we go beyond exploiting the existing algorithms and propose a novel subject-based triplet mining algorithm to address the challenge of feature extraction caused by subject heterogeneity and interclass similarities.

III. Methodology

A. Definitions

Human Activity Recognition task can be formulated as a time-series classification problem. More formally, given a sequence of signal vectors for T timestamps $\boldsymbol{X} = [\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_T}]$, where each vector $\boldsymbol{x_t} = [x_t^1, x_t^2, \ldots, x_t^S]$ consists of S channels, an objective is to predict a corresponding label y which is associated with a certain activity. While end-to-end deep learning models map input features directly onto the labels using a softmax output layer, a deep metric learning model might be considered as a function $f : \mathbb{R}^{S \cdot T} \rightarrow \mathbb{R}^D$ constructing feature embeddings, i.e. transform input features into the one-dimensional vector of size D.

This study focuses on models which use a triplet loss function, so-called triplet networks. To use them, input data points should be grouped into triplets $\{\boldsymbol{X}^a, \boldsymbol{X}^p, \boldsymbol{X}^n\}$ where the first two samples, namely anchor and positive, belong to one class and the remaining one, called negative, to another. The original triplet loss calculated for one triplet is defined as follows [15]:

$$L = ||f(\boldsymbol{X^a}) - f(\boldsymbol{X^p})||_2^2 - ||f(\boldsymbol{X^a}) - f(\boldsymbol{X^n})||_2^2 + \alpha, \quad (1)$$

where $||\cdot||_2^2$ refers to L_2-norm and α is a distance or violate margin. By minimizing the triplet loss function, the distance between the anchor and positive samples is also minimized, while the distance between the anchor and negative examples is maximized.

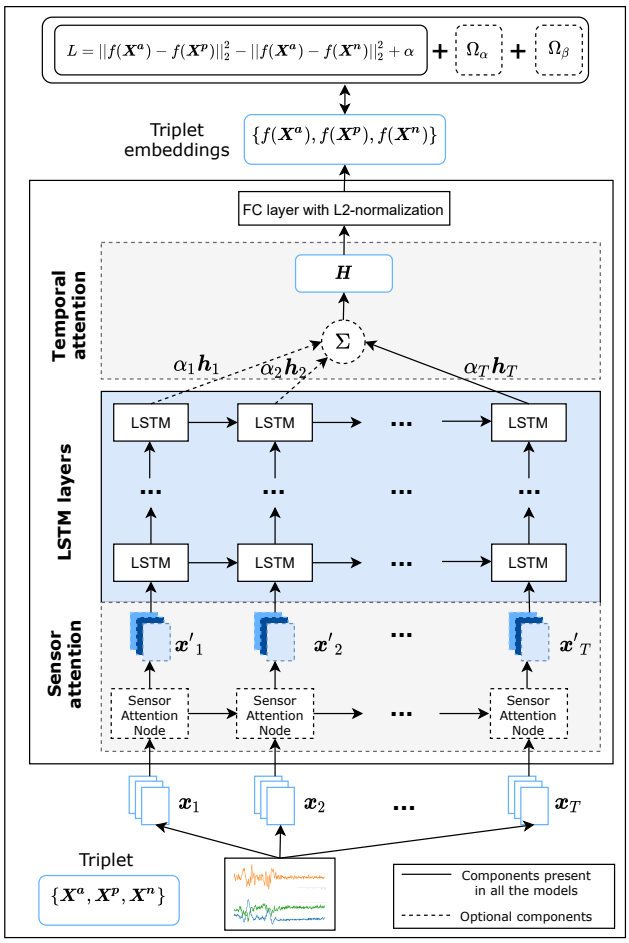


Fig. 2: Overview of the model architectures. The components which are required for all the models contain the LSTM layers and the fully-connected (FC) layer. The additional components which were designed to improve the performance of the baseline models are the sensor and temporal attention mechanisms.

B. Proposed Topology

In this paper, Long Short Term Memory (LSTM) networks are adapted to the deep metric learning approach and trained using triplet loss functions. The baseline triplet network consists of an LSTM layer which takes a vector $\boldsymbol{x_t} = [x_t^1, x_t^2, \ldots, x_t^S]$ as an input signal at timestep t. In total, there are T timesteps, thus, the shape of the input signals is $S \cdot T$. The output of the last time step is passed to a fully-connected layer producing embeddings of size D. We also applied L_2-normalization to the outputs of the last layer to obtain the final embeddings. The baseline triplet network architecture is illustrated in Fig. 1.

We upgrade the baseline by adapting temporal and sensor attention mechanisms for sensor-based HAR introduced in [13]. The schematic overview of the models is shown in Fig. 2. As can be seen from the figure, the proposed framework consists of three main blocks: sensor attention, LSTM layers and temporal attention. First, input signals are passed through sensor attention nodes. These nodes reweigh input signals

at each timestep according to their importance. Then, the weighted signals go through up to three LSTM layers. Outputs of LSTM layers from all the timesteps are sent to the temporal attention block in order to obtain the final LSTM output H. Finally, to get a feature embedding, vector H is passed through the fully-connected layer with L_2-normalization. After all examples in a triplet $\{\boldsymbol{X}^a, \boldsymbol{X}^p, \boldsymbol{X}^n\}$ are passed through all these steps, the triplet loss is calculated for the corresponding feature embeddings $\{f(\boldsymbol{X}^a), f(\boldsymbol{X}^p), f(\boldsymbol{X}^n)\}$. If attention mechanisms are implemented in their continuous forms, the loss also contains continuous regularization terms Ω_α for sensor attention and Ω_β for temporal attention (equations (9) and (8)). In the whole architecture, the attention blocks are optional and might be disabled. In total, the baseline model with no attention (n) and five models with different attention mechanisms were tried, namely: sensor attention (s), temporal attention (t), continuous sensor attention (cs), continuous temporal attention (ct) and both attentions in their continuous forms (csct).

C. LSTM and Attention Mechanisms

The LSTM network is a widely used type of Recurrent Neural Networks which can capture long-term dependencies better than a simple RNN [29]. Zeng et al. [13] proposed to use attention mechanisms, sensor and temporal attention, to enhance the predicting power of LSTMs with softmax output layers. Sensor attention is used to reweigh input sensor channels according to their importance, whereas temporal attention focuses on the most informative features extracted by the LSTM layer over time. More formally, given the input vector $\boldsymbol{x_t}$ at timestep t, the reweighed signal $\boldsymbol{x'_t}$ after applying sensor attention is calculated as [13]:

$$\boldsymbol{e_t} = \boldsymbol{w}_e^T tanh(\boldsymbol{W}_x x_t + \boldsymbol{W}_\beta \boldsymbol{\beta}_{t-1}), \tag{2}$$

$$\boldsymbol{\beta}_t = softmax(\boldsymbol{e_t}), \tag{3}$$

$$\boldsymbol{x'_t} = \boldsymbol{\beta}_t \odot \boldsymbol{x}_t, \tag{4}$$

where $\boldsymbol{w}_e^T$, $\boldsymbol{W}_x$, W_β are learnable parameters, $\boldsymbol{e_t}$ is an energy vector and $\boldsymbol{\beta}_t$ is a normalized energy vector. The attention weights are trained using gradient backpropagation, in the same way as other parameters of a model. It is clearly seen from equation (2) that the energy vector $\boldsymbol{e}_t$ depends on the vector $\boldsymbol{\beta}_{t-1}$ from the previous timestep. Hence, sensor attention passes information about the previous signals through time.

Unlike sensor attention, temporal attention does not contain recursive connections. It calculates the final output of the LSTM network by weighing hidden states from all the timesteps. The final output of the LSTM network with temporal attention $\boldsymbol{H}$ is calculated as follows [13]:

$$\boldsymbol{H} = \sum_{t=1}^{T} \alpha_t \boldsymbol{h}_t, \tag{5}$$

$$\alpha_t = \frac{exp\{score(\boldsymbol{h}_T, \boldsymbol{h}_t)\}}{\sum_{i=1}^{T} exp\{score(\boldsymbol{h}_T, \boldsymbol{h}_i)\}} \tag{6}$$

$$score(\boldsymbol{h}_T, \boldsymbol{h}_t) = \boldsymbol{h}_T^T \boldsymbol{W}_\alpha \boldsymbol{h}_t, \tag{7}$$

where α_t is an attention weight at timestep t, $\boldsymbol{W}_\alpha$ is the only learnable parameter and h_T is the output of an LSTM at the last timestep.

Finally, we also implemented a continuous version of both attention mechanisms as suggested in [13], which is a form of regularization restricting sharp differences of neighboring attention weights. Loss terms of the continuous regularization for temporal attention Ω_{temp} and sensor attention Ω_{temp} are calculated as follows:

$$\Omega_{temp}(\boldsymbol{\alpha}) = \lambda_1 \sum_{t=2}^{T} |\alpha_t - \alpha_{t-1}|, \tag{8}$$

$$\Omega_{sens}(\boldsymbol{\beta}) = \lambda_2 \sum_{t=2}^{T} |\boldsymbol{\beta_t} - \boldsymbol{\beta}_{t-1}|. \tag{9}$$

D. Triplet Sampling Methods

Informative triplet selection is a crucial aspect of training a triplet network. In this work, the emphasis is made on online triplet mining techniques, i.e. triplets are not pre-defined before training and, thus, are mined on the fly. We test two classical methods for triplet mining with the original triplet loss, namely hard and semi-hard sampling, as well as more specific and recent anchor-neighbor sampling with the Hierarchical Triplet Loss.

1) Hard and Semi-hard Triplets: The hard and semi-hard triplets are mined in an online manner, i.e. these triplets are selected based on distances between embeddings in a batch. The semi-hard triplets contain samples which are expressed by the following inequalities:

$$||f(\boldsymbol{X}^a) - f(\boldsymbol{X}^p)||_2^2 - ||f(\boldsymbol{X^a}) - f(\boldsymbol{X^n})||_2^2 < \alpha, \tag{10}$$

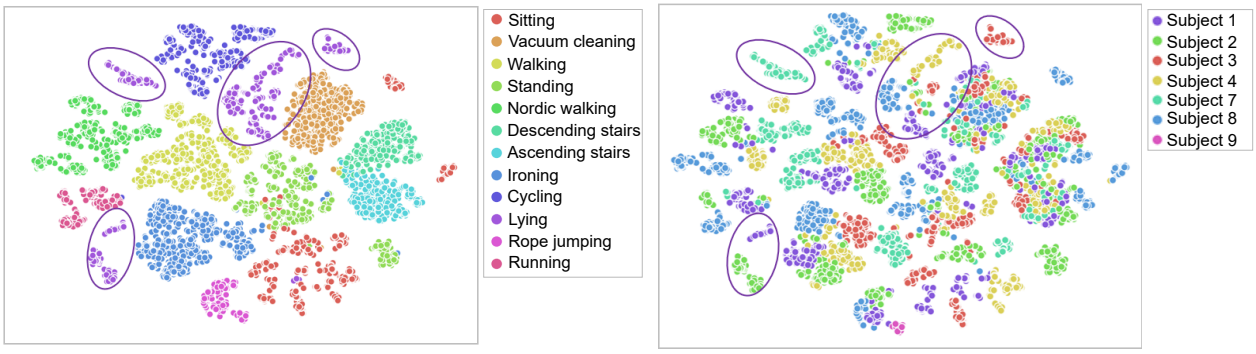
$$||f(\boldsymbol{X^a}) - f(\boldsymbol{X^p})||_2^2 < ||f(\boldsymbol{X^a}) - f(\boldsymbol{X^n})||_2^2. \tag{11}$$

This can be interpreted as finding those triplets for which the distance between the anchor ($f(\boldsymbol{X^a}$) and positive ($f(\boldsymbol{X^p})$) embeddings is smaller than the distance between the anchor and negative ($f(\boldsymbol{X^n})$) ones but the difference between these distances should not exceed the margin α [15]. The process of selecting data points for a mini-batch is random: K samples are selected for P random classes. The loss function for the mini-batch is computed as follows [15]:

$$L_{sh} = \sum_{i}^{N} max\Big\{0, ||f(\boldsymbol{X}_i^a) - f(\boldsymbol{X}_i^p)||_2^2 - ||f(\boldsymbol{X}_i^a) - f(\boldsymbol{X}_i^n)||_2^2 + \alpha\Big\}, \tag{12}$$

where N is the number of valid triplets which satisfy inequalities (10) and (11).

On the contrary, a triplet is classified as a hard one if the distance between the anchor and positive examples is larger than the distance between the anchor and negative samples [26]. Moreover, a widely used batch-hard strategy aims to calculate the loss on the most difficult triplets. Namely, for each anchor signal, X^a, the hardest positive and negative



(a) Colored activities

(b) Colored subjects

Fig. 3: Vanilla LSTM features visualization using t-SNE algorithm for the PAMAP2 dataset. The figure corresponds to the same features and has been colored in terms of activities (a) and subjects (b), in order to better illustrate the problems of inter-class similarities and person-dependent features. An activity for which subject heterogeneity has the obvious negative effect is lying. Clusters of data points corresponding to this activity are highlighted in purple ellipses in both figures. It is clearly seen from the right figure that most of the clusters of the activity belong to different subjects.

examples are sampled, in other words, from all valid triplets for the anchor signal, the most similar negative and the most different positive are selected. Finally, the loss function for the mini-batch can be formally written as [26]:

$$L_{bh} = \sum_{i}^{P}\sum_{a}^{K} max\Big\{0, \max_{p=1\ldots k}\{||f(\boldsymbol{X}_i^a) - f(\boldsymbol{X}_i^p)||_2^2\} - \min_{j=1\ldots p\neq i, n=1\ldots k}\{||f(\boldsymbol{X}_i^a) - f(\boldsymbol{X}_j^n)||_2^2\} + \alpha\Big\}. \tag{13}$$

2) Hierarchical Triplet Loss and Anchor-Neighbor Sampling: Unlike the original triplet loss with hard and semi-hard batch sampling, Hierarchical Triplet Loss (HTL) with Anchor-Neigbor (AN) sampling introduced in [14] exploits the global data distribution when mining triplets and calculating loss with a dynamic violate margin.

The main concept of the HTL approach is a hierarchical tree. The hierarchical tree is a specific data structure which stores information about distances between all pairs of classes in a dataset. It is constructed based on inter and intra-class distances. The inter-class distance for classes p and q is calculated as:

$$d_{inter}(p,q) = \frac{1}{n_p n_q}\sum_{i\in p, j\in q}||f(\boldsymbol{X}_i) - f(\boldsymbol{X}_j)||_2^2, \tag{14}$$

where n_p and n_q are the number of examples in classes p and q, respectively. The intra-class distance of class p is computed as follows:

$$d_{intra}(p) = \frac{1}{n_p^2 - n_p}\sum_{i\in p, j\in p}||f(\boldsymbol{X}_i) - f(\boldsymbol{X}_j)||_2^2, \tag{15}$$

Each leaf node of the tree corresponds to a certain class label. Then, two classes p and q are merged at a level l of the tree when the distance between these classes $d_{inter}(p,q)$ is smaller than a merging threshold d_l for this level. The merging threshold for level l is computed as follows:

$$d_l = \frac{l(4-d_0)}{L} + d_0, \tag{16}$$

where L is the maximum number of levels or depth of a tree. Hence, every pair of classes is merged at a certain level $l \in [1, L]$. In equation (16), d_0 refers to the zero-level threshold which is calculated as the average intra-class distance:

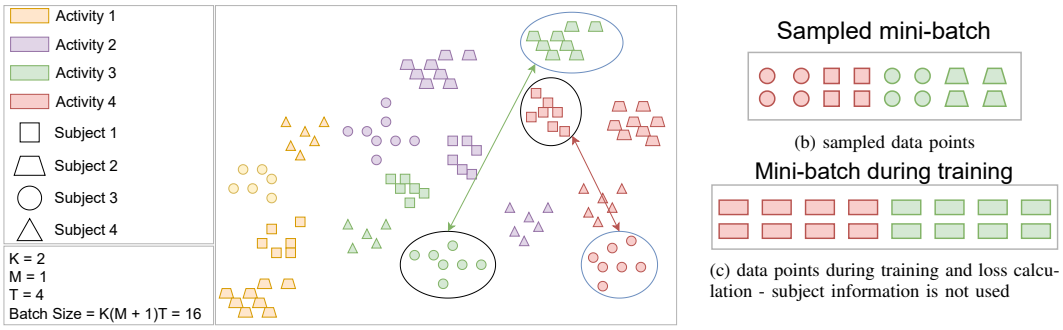
$$d_0 = \frac{1}{C}\sum_{c=1}^{C}\left(\frac{1}{n_c^2 - n_c}\sum_{i\in c, j\in c}||f(\boldsymbol{X}_i) - f(\boldsymbol{X}_j)||_2^2\right) = \frac{1}{C}\sum_{c=1}^{C} d_{intra}(c), \tag{17}$$

where n_c is the number of examples in class c and C is the number of classes in a dataset. The zero-level threshold is not used to merge classes and only needed to compute d_l.

Ge et al. [14] suggest pre-training a deep metric learning model using the original triplet loss so that data will not have the random distribution before training with HTL. The tree and distances are exploited to mine triplets using the anchor-neighbor sampling approach and calculate the dynamic margin for each positive-negative pair of classes in a triplet. The anchor-neighbor sampling algorithm benefits from the global data distribution stored in the tree and exploits it to mine triplets for a mini-batch. The first step of the anchor-neighbor sampling is to randomly select K classes. Then, for each of them, the M closest classes are sampled using the inter-class distances stored in the tree. Thus, since for each of K classes we sample M more neighboring ones, the total number of the selected classes is equal to $K \cdot (M+1)$. Finally, T data points are randomly extracted for each of the selected classes. As a result, the total number of data points in a mini-batch is $K \cdot (M+1) \cdot T$.

The novel term in HTL function is the dynamic violate margin: while in the original triplet loss the violate margin is fixed, the dynamic violate margin α_z is different for each pair of positive and negative classes and computed based on the hierarchical tree as follows:

$$\alpha_z = \beta + d_{l_{y_a, y_n}} - s_{y_a} \tag{18}$$



(a) A toy example setup

(b) sampled data points

(c) data points during training and loss calculation - subject information is not used

Fig. 4: A toy example of an iteration of the subject-based sampling. Assume, a dataset consists of 4 subjects and 4 activities, thus, the total number of activity-subject combinations is 16. Thus, the hierarchical tree has 4 nodes, one per activity. According to figure (a), K is set to 2, hence, 2 activity-subject combinations are selected randomly. Suppose, these are activity 3 from subject 3 and activity 4 from subject 1 (circled in black). Then, for each of them, the M farthest activity-subject combinations with the same activity but different subject labels are selected using distances between subclusters. In the figure, these are the second and third subjects for activities 3 and 4, respectively (circled in blue). As a result, $K \cdot (M+1) = 2 \cdot (1+1) = 4$ subclusters were selected for the mini-batch. Finally, from each of these subclusters, $T = 4$ data points should be taken randomly. Hence, as shown in figure (b), one batch contains 16 examples. However, during the process of training, the information about subjects is not used (figure (c)), so, the dynamic violate margin is calculated for activity classes based on the hierarchical tree.

where β is a constant, $d_{l_{y_a,y_n}}$ is the merging threshold (equation (16)) for the tree level at which anchor a and negative n classes were merged and s_{y_a} is the intra-class distance of the anchor class a computed according to equation (15). Finally, the value of the Hierarchical Triplet Loss on a mini-batch is calculated as follows:

$$L_h = \frac{1}{2N} \sum_{i}^{N} max\Big\{0, ||f(\boldsymbol{X}_i^a) - f(\boldsymbol{X}_i^p)||_2^2 - ||f(\boldsymbol{X}_i^a) - f(\boldsymbol{X}_i^n)||_2^2 + \alpha_z\Big\}, \tag{19}$$

where N is the number of triplets in the mini-batch. The detailed algorithm of anchor-neighbor sampling and training with HTL is available in [14].

The hierarchical tree is built before each epoch of training based on the training set embeddings. Hence, distribution of data is observed before a new epoch of training and stored in the hierarchical tree. It is important to mention that the hierarchical tree is not altered during the epoch and classes are not merged during training. In other words, the tree is only used to store information about inter-class distances, mine triplets and compute the dynamic violate margin for each pair of positive and negative examples in a mini-batch.

E. Subject-based sampling

Besides applying existing sampling techniques, we introduce a novel subject-based triplet mining algorithm which aligns with the HTL concept and exploits information about subjects and their similarities.

The proposed method aims to address the crucial problems of sensor-based HAR: subject heterogeneity and inter-class similarities. To illustrate these problems, we trained a simple model containing one Vanilla LSTM layer with a softmax output layer on the PAMAP2 dataset (described in Section IV-A) and visualized LSTM features in a two-dimensional space using the t-SNE algorithm (Fig. 3a). It can be clearly seen from the figure that some pairs of activities are not well separated. For example, the walking activity does not have a good separation from the Nordic walking activity (yellow and green clusters). Also, some activities do not form a single cluster and spread over the manifold as a group of subclusters. A good example is the laying activity (circled in purple). In Fig. 3b, we colored the same data points based on the subject information, i.e. each color in the right figure corresponds to a certain subject from the dataset. What is obvious is that the subclusters of the lying activity are obtained from different subjects. A similar situation is related to the sitting, Nordic walking and standing activities. The presence of such subject-based subclusters perfectly illustrates the problem of subject heterogeneity.

As for the original HTL algorithm, the suggested subject-based triplet mining approach is making use of the global data distribution while selecting triplets for each mini-batch. However, the major difference and novelty of the proposed approach is that it takes into account data instances belonging to the same activities which may differ significantly due to inter-subject differences. Specifically, we propose to include hard examples corresponding to the same activity class and different subjects into a mini-batch using the hierarchical tree and distances between them (Section III-D2). While sampling classes and data points for a mini-batch, we use distances between subclusters corresponding to activity-subject combinations. However, training cannot be done for these activity-subject combinations considered as classes because, in that case, a model treats data points corresponding to the same activities but different subjects as different classes and, thus, will push them apart from each other. It is crucial that only the original activities are considered as labels during training, i.e.

Algorithm 1: Training with Subject-based sampling

Input: Model f pre-trained using original triplet loss, training set $\{\boldsymbol{X}_i, s_i, y_i\}_{i=1}^{N}$, where s is a subject, y is an activity label and N is the number of data points, number of epochs n_e, sampling hyperparameters K, M and T.

while *epoch* $< n_e$ **do**

Compute distances between all activity-subject subclusters and build hierarchical tree H on Y activity classes;

$i = 0$*;*

while $i < N$ **do**

Randomly select K activity-subject combinations;

For each pair, select the M farthest activity-subject combinations with the computed distances.;

For each of $K \cdot (M+1)$ combinations, randomly select T data points;

Run the model on the mini-batch containing $K \cdot (M+1) \cdot T$ selected examples and compute the Hierarchical Triplet Loss L_h with dynamic violate margin computed based on tree H according to equation (19)*;*

Backpropogate gradients and update learnable parameters of model f;

i += $K \cdot (M+1) \cdot T$*;*

end

epoch $+= 1$*;*

end

the triplet loss is calculated for activity classes (not activity-subject combinations). Hence, the hierarchical tree is built based on activity labels and used for dynamic violate margin calculation (equation (18)) in the HTL function.

The complete algorithm for training a deep metric learning model with the subject-based sampling approach is described in Algorithm 1. At each iteration of the algorithm, the first step of sampling is a random selection of K activity-subject combinations from different activity classes. The next step is to find M farthest activity-subject combinations with the same activity label for each of the K combinations, using the distances between subclusters formed by activity-subject combinations. The last step of sampling is to randomly take T data points for the selected $K \cdot (M+1)$ activity-subject combinations. This will result in having a mini-batch of $K \cdot (M+1) \cdot T$ data points belonging to K original activity classes. A detailed toy example of an iteration of the subject-based sampling algorithm is shown in Fig. 4.

IV. Evaluations

A. Datasets

In this study, we use three open-source datasets which contain multi-activity data collected from various subjects. The pre-processing procedure is the same for all the datasets. First, long time-series chunks are segmented into overlapping time windows. Secondly, the datasets are normalized to zero mean and unit variance per channel. It is important to mention that test samples were segmented without overlapping.

PAMAP2. The PAMAP2 dataset [30] consists of 12 activities performed by 9 subjects. The activities in this dataset include basic movements, household activities and more unusual activities (e.g. rope jumping). Data is recorded at a frequency of 100 Hz using accelerometers, gyroscopes, magnetometers and heart rate sensors. We follow the recommendation of the dataset publishers to ignore the second accelerometer and in total use 28 channels. In this study, we use the typical pre-processing protocols from [7] and [13] so that we downsample data to 33.3 Hz and extract time windows of 5.12 seconds with 1-second overlap. We also use data obtained from subjects 5 and 6 as validation and test sets, respectively.

USC-HAD. The USC-HAD dataset [31] contains signals obtained from accelerometer and gyroscope sensors both having three degrees of freedom, i.e. 6 signals have been recorded per time step. In total, 14 subjects were involved in data collection performing 12 simple activities such as walking in different directions, running, jumping, etc. We use pre-processing and test protocols exploited in [20], i.e. the dataset was downsampled to 33.3 Hz, the length of time windows is 1 second with 50% overlapping and data from subjects 13 and 14 was used as a test set.

MHEALTH. The MHEALTH dataset [32], [33] was recorded using 3-axis accelerometers, gyroscopes and magnetometers placed on chest, ankles and arms. The dataset consists of measurements collected from 10 subjects while performing 12 activities. For this dataset, we keep the initial frequency of data (50 Hz), use time windows of 1 second with 50% overlapping and exploit data from subjects 9 and 10 as validation and test sets, respectively.

B. Implementation Details

The models implemented in this paper are described in Section III-B and shown in Fig. 2. The hyperparameter tuning was performed on the validation sets of the datasets. All the hyperparameter values tried in the experiments are summarized in Table I. In the table, the optimal set of hyperparameters is highlighted for each dataset. Parameters of the models were optimized using Stochastic Gradient Descent with Adaptive Moment Estimation (ADAM) with the default parameters ($\epsilon = 10^{-8}$,$\beta_1 = 0.9$, $\beta_2 = 0.999$) [34]. All the models were trained for 100 epochs. The learning rate is decreased twice after each 10 epochs period with no performance improvement. Training with HTL requires pre-training with the original triplet loss, so the best models on validation sets were fine-tuned with HTL for 10 more epochs. For all our models including the LSTM baseline trained with the original triplet loss, we fix the size of the LSTM hidden state vectors to 512. For all the triplet networks, we kept the size of a mini-batch close to 128. Specifically, we fixed hyperparameters $P = 6$ and $K = 20$ for the original triplet

	Attention type	Number of layers	Embedding size	Violate margin	Initial learning rate	Triplet sampling method
Original Triplet Loss	n, s, cs, **t**, ct, csct	1, 2, **3**	256, **512**, 1024	**0.2**, 0.5, 1.0	10^{-3}, 10^{-4}, $\mathbf{10^{-5}}$	**hard**, semi-hard
Hierarchical Triplet Loss	n, s, cs, **t**, ct, csct	1, 2, **3**	256, **512**, 1024	dynamic	10^{-3}, 10^{-4}, $\mathbf{10^{-5}}$	**anchor-neighbor**, subject-based

TABLE I: Hyperparameters used for training. The attention types are encoded as in Section IV-B. The optimal hyperparameters for the PAMAP2 dataset are highlited in bold; for USC-HAD - underlined; for MHEALTH - in gray boxes.

Model	PAMAP2			USC-HAD			MHEALTH		
Baselines									
Softmax LSTM baseline	0.792			0.49			0.442		
Triplet LSTM baseline (ours, Fig. 1)	0.811			0.535			0.376		
Recent benchmarks									
DeepConvLSTM [9]	0.748			0.46			-		
b-LSTM-S [7]	0.868			-			-		
LSTM with continuous sensor and continuous temporal attention [13]	0.8996			0.553			0.48		
Transformer-like architecture [20]	different pre-processing			0.55			-		
Proposed Triplet Networks with the best performance (ours)									
	OTL	HTL-AN	HTL-SB	OTL	HTL-AN	HTL-SB	OTL	HTL-AN	HTL-SB
Triplet LSTM with temporal attention	**0.904**	0.893	0.884	0.54	0.581	0.596	0.648	0.641	**0.656**
Triplet LSTM with continuous temporal attention	0.902	0.874	0.881	0.612	0.592	**0.628**	0.55	0.504	0.519

TABLE II: F1-scores of the best models trained with the triplet loss along with the baselines and recent benchmarks which used the same training and evaluation protocols. For the proposed models, we report three values per architecture: Original Triplet Loss (OTL), HTL with anchor-neighbor sampling (HTL-AN) and HTL with subject-based sampling (HTL-SB). Note: - means that results are not available.

loss (described in Section III-D1), and $K = 4$, $M = 2, T = 10$ for the HTL (Sections III-D2 and III-E).

The experiments were conducted on NVIDIA Titan X GPU with 12 GB memory. It is important to note that Deep Metric Learning (DML) models mighty require more time for training than softmax-based models. For example, using Titan X on pre-processed data of the PAMAP2 dataset, one epoch of training DML models with subject-based sampling and HTL takes about 5 minutes. On the other hand, an epoch of training softmax-based models with attention takes 2 minutes. During inference and classification, the evaluation of DML models using the extracted training embeddings is efficient. Moreover, classification methods such as K-Nearest Neighbour (KNN) can be employed on the training and validation embeddings without the need for a GPU.

C. Evaluation Protocols and Experimental Setup

The proposed models generate feature representations of the input signals. Thus, it is required to use another algorithm which will map feature embeddings onto activity labels. Since DML models aim to group the data points corresponding to the same class together and separate different classes in the embedding space, the kNN algorithm which makes use of distances between data points is used in this study as a classifier for the obtained feature embeddings. Based on the conducted experiments, we noticed that the value of k does not affect model performance significantly, so we fixed $k = 7$, which is optimal in terms of metrics and computational costs.

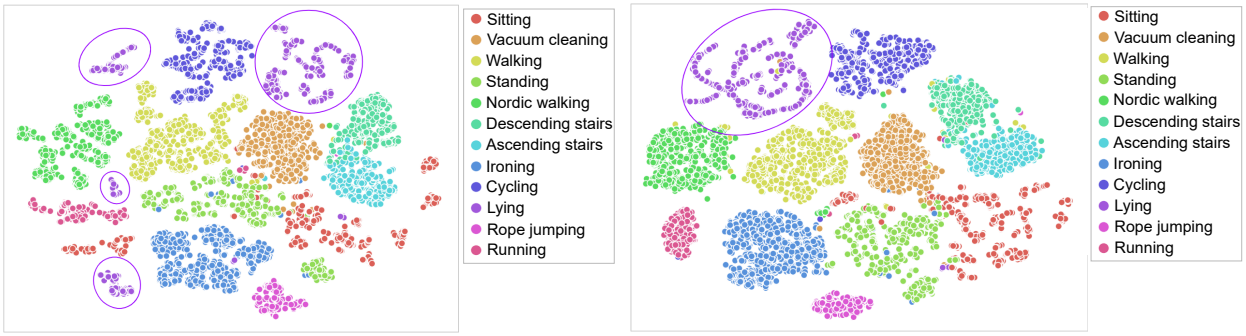
After each epoch of training, the models are assessed on a validation set of a dataset. The model weights are saved if the model has improved its performance. Therefore, we test the model with weights shown the best performance on the validation set. The performance of the models is assessed using the average macro F1-score.

D. Results and Baseline Comparisons

In this study, we compare the proposed DML models against recent architectures based on the traditional end-to-end learning approach which follow the same pre-processing steps and evaluation protocols for the selected datasets. Table II aggregates the values of F1-scores corresponding to the baselines, recent benchmarks and the proposed models applied to the PAMAP2, USC-HAD and MHEALTH datasets. It is crucial to mention that some of the works used different pre-processing and validation protocols (e.g. Transformer-like architecture presented in [20] for the PAMAP2 dataset) or different set of labels (such as InnoHAR [10] for the PAMAP2 dataset). The results in such cases were not included into the table. In addition, we evaluated the best LSTM model with continuous sensor and continuous temporal attention from [13] on the MHEALTH and USC-HAD datasets with hyperparameters suggested in the paper in order to compare its performance against our triplet models.

First, the baseline model, illustrated in Fig. 1, outperformed the end-to-end Vanilla LSTM baseline by about 2% on PAMAP2 and 4% on USC-HAD, however, it fell behind on the MHEALTH dataset. More importantly, we improved the state-of-the-art results on all the datasets using the proposed triplet networks paradigm. Specifically, triplet models with temporal attention have shown the best performance among all the topologies with attention. For instance, the DML model with temporal attention has the highest F1-score on the PAMAP2 and MHEALTH datasets while the triplet LSTM with continuous temporal attention outperformed all the benchmarks on the USC-HAD dataset.

Interestingly, fine-tuning with the HTL function and the proposed subject-based sampling algorithm significantly improve the performance of the triplet networks on the USC-HAD (about 8%) and MHEALTH (more than 17%) dataset



(a) Original triplet loss with batch-hard sampling

(b) Hierarchical triplet loss with subject-based mining

Fig. 5: Visualization of feature embeddings for the PAMAP2 training set obtained by the best models trained with the triplet loss. The subject-based mining method integrated into the HTL concept improves the structure of activity clusters.

	lying	sitting	standing	walking	running	cycling	Nordic walking	ascending stairs	descending stairs	vacuum cleaning	ironing	rope jumping	average distance
Original Triplet Loss	0.115	0.168	0.17	0.162	0.156	0.184	0.165	0.204	0.218	0.219	0.185	0.175	0.177
HTL with subject-based sampling	0.065	0.086	0.096	0.078	0.125	0.079	0.122	0.109	0.114	0.186	0.095	0.097	0.104

TABLE III: Intra-class distances for PAMAP2 activities clusters produced by the triplet networks shown the best performance.

comparing to the presented benchmark models. Although the models trained with HTL and subject-based sampling have not outperformed the one trained with the original triplet loss on the PAMAP2 dataset, the proposed algorithm improved activities' representations, as elaborated in Section IV-E. The obtained results support the hypothesis that it is possible to benefit from subject information and triplets containing data points obtained from subjects with different behaviour patterns are informative and contribute to model training.

E. Feature extraction and feature embeddings visualization

The major advantage of metric learning is that, instead of learning activity classes directly, models learn effective feature representations in a manifold space. Hence, the DML models can be particularly useful for feature extraction.

Fig. 5 demonstrates feature representations of the PAMAP2 training set data points in a two-dimensional space using the t-SNE algorithm. The left part of the figure is obtained for the best model (with temporal attention) trained with the original triplet loss and batch-hard sampling strategy. It is clearly seen that class separation is more explicit as well as individual activities are better grouped than for the LSTM baseline with softmax output (Fig. 3a). However, subject heterogeneity still affects the quality of the obtained feature embeddings since several activities (lying, sitting, Nordic walking) are still represented as the number of clusters spread over the manifold. Fig. 5b shows the embeddings obtained using the Hierarchical Triplet Loss with the proposed subject-based sampling algorithm. It is obvious that almost all the classes are grouped visually better, although the F1-score for the model is lower (Table II). In particular, the lying activity data points (in purple) which were spread over the manifold space for both softmax (Fig. 3a) and original triplet loss (Fig. 5a) models are now concentrated in one area of the feature space. This shows that the proposed subject-based triplet mining approach has a potential to address the subject heterogeneity problem.

Furthermore, in order to support the findings from embeddings visualization, intra-class distances were computed according to equation (15) for each activity cluster in the PAMAP2 dataset for both best models: trained with original and hierarchical triplet loss functions (shown in Table III). According to the table, all intra-class distances for the model trained using hierarchical triplet loss and subject-based triplet mining are lower than for the original triplet model. As a result, the average intra-class distance is decreased by about 40%. These results also illustrate that the proposed subject-based triplet mining algorithm has a positive effect on features extracted by triplet networks and attempts to address the problems of subject heterogeneity and inter-class similarities.

V. Conclusion

In this paper, we address the major problems of sensor-based HAR, namely subject heterogeneity and inter-activity similarity. This is done by adapting triplet networks to sensor-based HAR and proposing a novel subject-based triplet sampling algorithm which uses information about subject similarities while mining informative triplets. The proposed LSTM-based baseline triplet model was upgraded by attention mechanisms. The results of the experiments have shown that triplet networks not only improve the quality of recognition but also are capable of constructing robust feature representations less affected by subject heterogeneity and inter-class similarities. The future work on deep metric learning for sensor-based HAR might include experiments with different encoders, such as Transformers, as well as more thorough evaluations of classifiers built on top of the extracted feature embeddings.

Acknowledgement

This work has been funded by the European Union's Horizon2020 project: PeRsOnalized Integrated CARE Solution for Elderly facing several short or long term conditions and enabling a better quality of LIFE (PROCareLife), under Grant Agreement N.875221.

REFERENCES

[1] M. Panwar, D. Biswas, H. Bajaj, M. Jobges, R. Turk, K. Maharatna, and A. Acharyya, "Rehab-net: Deep learning framework for arm movement classification using wearable sensors for stroke rehabilitation," *IEEE Transactions on Biomedical Engineering*, vol. PP, pp. 1–1, 02 2019.

[2] P. Skocir, P. Krivic, M. Tomeljak, M. Kusek, and G. Jezic, "Activity detection in smart home environment," *Procedia Computer Science*, vol. 96, pp. 672–681, 12 2016.

[3] R. Grzeszick, J. M. Lenk, F. M. Rueda, G. A. Fink, S. Feldhorst, and M. ten Hompel, "Deep neural network based human activity recognition for the order picking process," in *Proceedings of the 4th International Workshop on Sensor-Based Activity Recognition and Interaction*, ser. iWOAR '17. New York, NY, USA: Association for Computing Machinery, 2017. [Online]. Available: https://doi.org/10.1145/3134230.3134231

[4] 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," *ArXiv*, vol. abs/2001.07416, 2020.

[5] J. B. Yang, M. N. Nguyen, P. P. San, X. L. Li, and S. Krishnaswamy, "Deep convolutional neural networks on multichannel time series for human activity recognition," in *Proceedings of the 24th International Conference on Artificial Intelligence*, ser. IJCAI'15. AAAI Press, 2015, p. 3995–4001.

[6] A. Ignatov, "Real-time human activity recognition from accelerometer data using convolutional neural networks," *Applied Soft Computing*, vol. 62, pp. 915 – 922, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494617305665

[7] N. Y. Hammerla, S. Halloran, and T. Plötz, "Deep, convolutional, and recurrent models for human activity recognition using wearables," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, ser. IJCAI'16. AAAI Press, 2016, p. 1533–1540.

[8] Y. Zhao, R. Yang, G. Chevalier, X. Xu, and Z. Zhang, "Deep residual bidir-lstm for human activity recognition using wearable sensors," *Mathematical Problems in Engineering*, vol. 2018, p. 7316954, Dec 2018. [Online]. Available: https://doi.org/10.1155/2018/7316954

[9] F. Ordóñez and D. Roggen, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, p. 115, Jan 2016. [Online]. Available: http://dx.doi.org/10.3390/s16010115

[10] C. Xu, D. Chai, J. He, X. Zhang, and S. Duan, "Innohar: A deep neural network for complex human activity recognition," *IEEE Access*, vol. 7, pp. 9893–9902, 2019.

[11] H. Ma, W. Li, X. Zhang, S. Gao, and S. Lu, "Attnsense: Multi-level attention mechanism for multimodal human activity recognition," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, 7 2019, pp. 3109–3115. [Online]. Available: https://doi.org/10.24963/ijcai.2019/431

[12] J. V. Jeyakumar, L. Lai, N. Suda, and M. Srivastava, "Sensehar: A robust virtual activity sensor for smartphones and wearables," in *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*, ser. SenSys '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 15–28. [Online]. Available: https://doi.org/10.1145/3356250.3360032

[13] M. Zeng, H. Gao, T. Yu, O. J. Mengshoel, H. Langseth, I. Lane, and X. Liu, "Understanding and improving recurrent networks for human activity recognition by continuous attention," in *Proceedings of the 2018 ACM International Symposium on Wearable Computers*, ser. ISWC '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 56–63. [Online]. Available: https://doi.org/10.1145/3267242.3267286

[14] W. Ge, W. Huang, D. Dong, and M. R. Scott, "Deep metric learning with hierarchical triplet loss," in *Computer Vision – ECCV 2018*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham: Springer International Publishing, 2018, pp. 272–288.

[15] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 815–823.

[16] W. Chen, X. Chen, J. Zhang, and K. Huang, "Beyond triplet loss: A deep quadruplet network for person re-identification," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1320–1329.

[17] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*, ser. CVPR '06. USA: IEEE Computer Society, 2006, p. 1735–1742. [Online]. Available: https://doi.org/10.1109/CVPR.2006.100

[18] L. van der Maaten and G. Hinton, "Viualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 11 2008.

[19] W. Jiang and Z. Yin, "Human activity recognition using wearable sensors by deep convolutional neural networks," in *Proceedings of the 23rd ACM International Conference on Multimedia*, ser. MM '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 1307–1310. [Online]. Available: https://doi.org/10.1145/2733373.2806333

[20] S. Mahmud, M. T. H. Tonmoy, K. Bhaumik, A. Rahman, M. A. Amin, M. Shoyaib, M. Khan, and A. Ali, "Human activity recognition from wearable sensor data using self-attention," 02 2020.

[21] A. Bulling, U. Blanke, and B. Schiele, "A tutorial on human activity recognition using body-worn inertial sensors," *ACM Comput. Surv.*, vol. 46, no. 3, Jan. 2014. [Online]. Available: https://doi.org/10.1145/2499621

[22] S.-A. Rokni, M. Nourollahi, and H. Ghasemzadeh, "Personalized human activity recognition using convolutional neural networks," *AAAI 2018*, 01 2018.

[23] E. Soleimani and E. Nazerfard, "Cross-subject transfer learning in human activity recognition systems using generative adversarial networks," *CoRR*, vol. abs/1903.12489, 2019. [Online]. Available: http://arxiv.org/abs/1903.12489

[24] K. Chen, L. Yao, D. Zhang, X. Chang, G. Long, and S. Wang, "Distributionally robust semi-supervised learning for people-centric sensing," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 3321–3328, 07 2019.

[25] H. O. Song, Y. Xiang, S. Jegelka, and S. Savarese, "Deep metric learning via lifted structured feature embedding," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 4004–4012.

[26] A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re-identification," *ArXiv*, vol. abs/1703.07737, 2017.

[27] M. Abdu-Aguye and W. Gomaa, "Robust human activity recognition based on deep metric learning," in *ICINCO (1)*, 01 2019, pp. 656–663.

[28] T. Sheng and M. Huber, "Siamese networks for weakly supervised human activity recognition," in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, 2019, pp. 4069–4075.

[29] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, p. 1735–1780, Nov. 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

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

[31] M. Zhang and A. Sawchuk, "Usc-had: a daily activity dataset for ubiquitous activity recognition using wearable sensors," 09 2012, pp. 1036–1043.

[32] O. Banos, R. Garcia, J. A. Holgado-Terriza, M. Damas, H. Pomares, I. Rojas, A. Saez, and C. Villalonga, "mhealthdroid: A novel framework for agile development of mobile health applications," in *Ambient Assisted Living and Daily Activities*, L. Pecchia, L. L. Chen, C. Nugent, and J. Bravo, Eds. Cham: Springer International Publishing, 2014, pp. 91–98.

[33] O. Banos, C. Villalonga, R. García, A. Saez, M. Damas, J. Holgado-Terriza, S. Lee, H. Pomares, and I. Rojas, "Design, implementation and validation of a novel open framework for agile development of mobile health applications," *BioMedical Engineering OnLine*, vol. 14, p. S6, 08 2015.

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2014.