Unsupervised Deep Learning-based clustering for Human Activity Recognition

Hamza Amrani, Daniela Micucci and Paolo Napoletano

Department of Informatics, Systems and Communication

University of Milano - Bicocca , Milan, Italy

Email: h.amrani@campus.unimib.it, daniela.micucci@unimib.it, paolo.napoletano@unimib.it

Abstract—One of the main problems in applying deep learning techniques to recognize activities of daily living (ADLs) based on inertial sensors is the lack of appropriately large labeled datasets to train deep learning-based models. A large amount of data would be available due to the wide spread of mobile devices equipped with inertial sensors that can collect data to recognize human activities. Unfortunately, this data is not labeled. The paper proposes DISC (Deep Inertial Sensory Clustering), a DL-based clustering architecture that automatically labels multi-dimensional inertial signals. In particular, the architecture combines a recurrent AutoEncoder and a clustering criterion to predict unlabelled human activities-related signals. The proposed architecture is evaluated on three publicly available HAR datasets and compared with four well-known end-to-end deep clustering approaches. The experiments demonstrate the effectiveness of DISC on both clustering accuracy and normalized mutual information metrics.

Index Terms—Human Activity Recognition, Unsupervised learning, Deep learning, AutoEncoder

I. INTRODUCTION AND BACKGROUND

Automated recognition of activities of daily living (ADLs) based on inertial data is popular in various fields, such as surveillance, health care, and delivery [1]–[3]. Nowadays, supervised classification of ADLs is mainly based on deep learning (DL) approaches, which are well known to be data hungry [4]. Many labeled datasets are available in the literature, however to scale-up the use of these methods in several domains, a very large amount of data is required [5].

One way to speed up the labeling of ADL samples is to take advantage of the clustering analysis that is an unsupervised strategy to group similar data into clusters [6]. Its applications include automatic data labeling for supervised learning and pre-processing for data visualization and analysis.

Most common methods consist of a two-stages process: *stage 1)* a deep neural network (DNN) procedure for learning deep representative features and *stage 2)* a Machine-Learning (ML)-based clustering algorithm for data grouping [7]–[9].

Deep Embedded Clustering (DEC) [10] learns a mapping from the observed space to a low-dimensional latent space leveraging Stacked AutoEncoders (SAE), which can obtain feature representations and cluster assignments simultaneously.

Improved Deep Embedded Clustering (IDEC) [11] is a modified version of DEC that incorporates an under-complete AutoEncoder to preserve the local structure. The preservation of the local structure prevents the embedded space from being distorted by fine-tuning, thus ensuring embedded spatial features' representativeness.

Deep Convolutional Embedded Clustering (DCEC) [12] improves IDEC by replacing Stacked AutoEncoders (SAE) with Convolutional AutoEncoders(CAE) to preserve the local data structure.

Recently, Balanced Deep Embedded Clustering (BDEC) [13] has been proposed to address the inherent vulnerability of DEC to data imbalance by utilizing a pre-processing step that uses a scalable representative algorithm to extract a balanced subset and use it for training.

Finally, Deep Sensory clustering (DSC) [6] is a recent clustering architecture that employs a Recurrent AutoEncoder (RAE) and a clustering criterion to learn clustering-friendly representations from inertial sensory readings. DSC can be considered the first approach using deep clustering on inertial signals.

The paper proposes Deep Inertial Sensory Clustering (DISC), a deep-learning-based clustering architecture that automatically performs unsupervised learning and label annotation by analyzing multi-dimensional inertial signals. The architecture combines a recurrent AutoEncoder and a clustering criterion to predict unlabelled human activities-related signals. As for previous approaches in the state of the art, our architecture consists of two parts. The first part covers stage 1. It consists of an AutoEncoder that tries to reconstruct two outputs from the input signals: the inverse input and the future sequence of the input. The second part covers stage 2 and is devoted to clustering the dimensionally reduced signals from stage 1.

The main contribution of the paper with respect to the state of the art is the use of ConvGRU layers (Convolutional Gate Recurrent Unit [14]) that allow spatio-temporal features to be extracted. The recursive layers (LSTM, GRU) have been shown to be useful in extracting temporal features, while the convolutional layers in extracting spatial features.

The achieved architecture has been compared with four state-of-the-art techniques (DEC [10], IDEC [11], DCEC [12], and DCS [6]) on three publicly available HAR datasets (UCI HAR [15], Skoda [16], and MHEALTH [17]). Results show, on average, its effectiveness in the inertial sensory clustering task.

The paper is organized as follows. Section II introduces the architecture; Section III describes the implementation details

of the architecture, the metrics used to validate the approach, and the datasets used in the evaluation. SectionIV discusses the results obtained. Finally Section V presents the conclusions and the future directions.

II. ARCHITECTURE

This section presents our proposal Deep Inertial Sensory Clustering (DISC). The neural architecture of our proposal is sketched in Figure 1. The first subsection shows the architecture used to implement stage 1 (i.e., the Multi-Task AutoEncoder employed), and the second the architecture for stage 2 (i.e., the Clustering Criterion adopted).

A. Stage 1: Multi-Task AutoEncoder

The architecture leverages a *Recurrent Neural Network (RNN)-based autoencoder (RNN-AE)* since RNN architectures have been proved to be powerful for handling sequential data, such as text, sounds, and HAR signals [18], [19].

The RNN-AE architecture consists of three RNNs: a recurrent encoder ConvGRU [14] (Enc_θ) and two conditional decoders GRU [20] (Dec_ϕ).

The input is a window x of length T which contains $x_1, ..., x_T$ elements. Each x_t element includes one or more values according to the number of channels N.

1) Recurrent Encoder (Enc_θ): The recurrent encoder Enc_θ learns a compacted representation of the spatio-temporal features that characterizes the activities of daily living (ADLs).

In particular, a bi-directional Convolutional Gated Recurrent Unit (ConvGRU) [14][1] reads a windows x in both forward and backward directions and updates its hidden internal state for each $x_t \in x$.

Equations from 1 to 4 define the update rule.

$$g_t = \sigma(W_g \star_m [h_{t-1}; x_t] + b_g) \qquad (1)$$

$$r_t = \sigma(W_r \star_m [h_{t-1}; x_t] + b_r) \qquad (2)$$

$$\tilde{h}_t = tanh(W_c \star_m [x_t; g_t \odot h_{t-1}] + b_c) \qquad (3)$$

$$h_t = (1 - r_t) \odot \tilde{h}_t + r_t \odot h_{t-1} \qquad (4)$$

where h_t is the output for x_{t-1}; g_t and r_t are the *update* and the *forget* gates respectively, σ is the *sigmoid* function, $\star_m$ represents a *convolution* with a kernel of size m x m, b's are *bias terms*, $\odot$ denotes *element-wise multiplication (or Hadamard product)*, and $W_{g,r,c}$ are 2D-convolutional kernels.

After processing the entire sequence x, the final hidden state h_T of 512 dimensions is reduced through a fully connected layer to 256 dimensions.

The resulting low-dimensional feature $z \in \mathbb{R}^d$ of equation 5 encodes information about the ADL through a representation of the spatio-temporal dependencies that are present in the input sequence x. The size of the feature space is $d = 256$.

$$z = Enc_\theta(x) \qquad (5)$$

The hidden states of the convolutional GRU is initialized with zeros.

[1]A RNN that combines Gated Recurrent Units (GRUs) [20] with the convolution operation.

2) Conditional Recurrent Decoders (Dec_ϕ): The decoder is based on gated GRU and it is in charge of reconstructing back the input sequence x from z by using a multi-task strategy that includes two different decoders Dec_ϕ. This strategy minimizes two different Mean Square Error (MSE) losses: one to optimize the reconstruction of a time-flipped version of the input sequence, and the other to optimize the reconstruction of a future (anticipated) sequence.

In the proposed architecture, both recurrent decoders are conditional. In fact, the decoder GRU needs two inputs at stage t: the input sequence x_t and the hidden state h_t. At the decoder stage we do not have an input sequence and thus the input x_t is initialized, for $t = 0$, with zeros. Differently, the hidden state is initialized at stage $t = 0$ with a transformation of the embedding z. This transformation, named *context vector*, is achieved through a convolutional layer that maps a 256-dimensional embedding in a 512-dimensional one. At stage $t > 0$ the input $x_t = h_{t-1}$ while h_t is the hidden state at previous stage that is updated by the training procedure.

This multi-task strategy, along with the conditional initialization of the hidden states, forces the AutoEncoder to learn a meaningful representation (embedding space) where sequences belonging to different classes are well separated in the representation space.

Equations from 6 to 9 define the activation h_t.

$$g_t = \sigma(W_g x_t + U_g h_{t-1}) \qquad (6)$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1})) \qquad (7)$$

$$\tilde{h}_t = tanh(W x_t + U(r_t \odot h_{t-1})) \qquad (8)$$

$$h_t = (1 - g_t)h_{t-1} + g_t \tilde{h}_t \qquad (9)$$

Summing up, the output of the decoder is:

$$(\bar{y}^{\,rec}, \bar{y}^{\,fut}) = Dec_\phi(z) \qquad (10)$$

where $\bar{y}^{\,rec}$ and $\bar{y}^{\,fut}$ are the reconstructed and the anticipated sequences generated from the input x.

The *objective* of the Recurrent AutoEncoder is a joint objective function:

$$L_{AE} = L_{rec} + L_{fut} =$$
$$\left\| y^{\,rec} - \bar{y}^{\,rec} \right\|^2 + \left\| y^{\,fut} - \bar{y}^{\,fut} \right\|^2 \qquad (11)$$

where L_{rec} and L_{fut} indicate the reconstruction loss and the future prediction loss, respectively, and denote the mean square errors between decoder's generated output sequences ($\bar{y}^{\,rec}$ and $\bar{y}^{\,fut}$) and the expected target sequences ($y^{\,rec}$ and $y^{\,fut}$).

The *optimal network parameters* of encoder $z = Enc_\theta(x)$ and decoder $(\bar{y}^{\,rec}, \bar{y}^{\,fut}) = Dec_\phi(z)$ are updated by minimizing the reconstruction error:

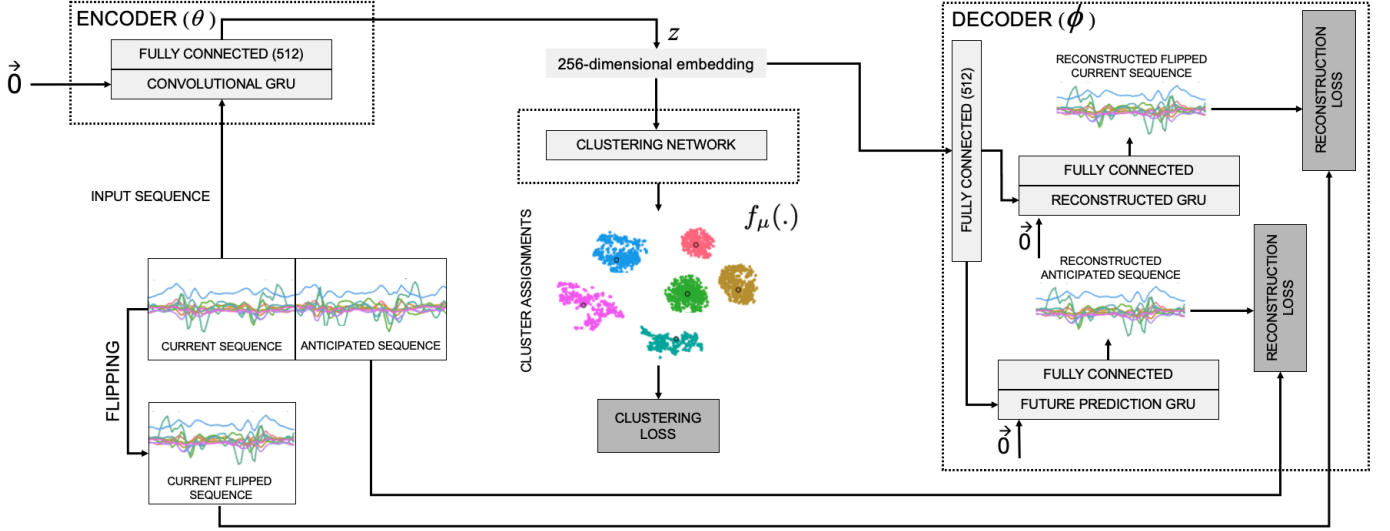$$(\theta^*, \phi^*) = min_{\theta,\phi} L_{AE} \qquad (12)$$

Fig. 1: Deep Learning-based Inertial Sensor Clustering architecture.

B. Stage 2: Clustering Criterion

A parameterized clustering network $f_\mu(.)$ is connected to the AutoEncoder embedding layer to estimate the cluster assignment distributions and to map each z into a soft label.

It uses the similarities between the data representations and cluster centroids $\{\mu_j\}_{j=1}^k$ to compute soft cluster assignments, while its loss enforces the soft assignments to have more stringent probabilities. The initial cluster centroids are obtained from classical Clustering algorithms on the embedded representations $z = Enc_\theta(x)$ after the pre-training of Multi-Task AutoEncoder, and are initialized only once at the beginning of the refinement stage.

The clustering network $f_\mu(.)$ mantains cluster centroids $\{\mu_j\}_{j=1}^k$ as trainable weights and maps a set of m embedded points $\{z_i \in Z\}_{i=1}^m$ into soft label $Q_i = f_\mu(z_i) = (q_{ij})_{j=1}^k$ by following the Student's t-distribution [21]:

$$q_{ij} = \frac{(1 + \|z_i - \mu_j\|^2)^{-1}}{\sum_{j'=1}^k (1 + \|z_i - \mu_{j'}\|^2)^{-1}} \qquad (13)$$

where q_{ij} is the j-th entry of q_i, which represents the probability of z_i belonging to cluster j.

By squaring this distribution and then normalizing it, the auxiliary distribution $P_i = (p_{ij})_{j=1}^k$ [10] forces assignments to have stricter probabilities (i.e., closer to 0 and 1). P_i helps to improve cluster purity, emphasizing on data points assigned with high confidence, and to prevent large clusters from distorting the hidden feature space.
It is defined as:

$$p_{ij} = \frac{q_{ij}^2 / \sum_i^m q_{ij}}{\sum_{j'=1}^k (q_{ij'}^2 / \sum_{i=1}^m q_{ij'})} \qquad (14)$$

The *Cluster Assignment Hardening (CAH)* loss L_C is defined through minimizing the *Kullback-Leibler (KL) diver-*

gence [22] between the distribution of soft labels and the auxiliary target distribution, as:

$$L_C = KL(P\|Q) = \sum_i^m \sum_j^k p_{ij}\, log\frac{p_{ij}}{q_{ij}} \qquad (15)$$

Lower the KL divergence value, the better we have matched the true distribution with our approximation.

The reconstruction loss of the AutoEncoder is joined to the objective and optimized along with the Cluster Assignment Hardening loss simultaneously, preserving the local structure of data generating distribution and avoiding the corruption of feature space. The final joint optimization criterion is:

$$L = \gamma L_C + L_{AE} \qquad (16)$$

where the coefficient $\gamma \in [0,1]$ controls the Clustering objective contribution. The optimal network parameters are optimized with respect to the global criterion as:

$$(\theta^*, \phi^*, \omega^*) = min_{\theta,\phi,\mu}\, L \qquad (17)$$

III. MATERIAL AND METHODS

A. Network Architecture

The recurrent encoder Enc_θ is a 2-layer bi-directional ConvGRU with 256 hidden states and the conditional recurrent decoders Dec_ϕ use uni-directional connections with 512 hidden states. The bottleneck embedded dimension of autoencoders is set to 256.

A single layer in the clustering network $f_\mu(.)$ has been used to integrate the CAH (Clustering Assignment Hardening).

All experiments share the same network architecture.

B. Optimization Settings

The recurrent AutoEncoder is pre-trained end-to-end in mini-batches of size 256 for 100 epochs using the Adam

optimizer [23]. The initial learning rate has been set to 10 and, after 70 epochs, decayed by a factor of 10.

A Batch Normalization [24] operation is used in the encoder fully connected layer to make the training faster and more stable. Then, the recurrent AutoEncoder is refined with the clustering objective till the cluster assignment changes among two consecutive epochs are less than 0.1%.

The coefficient γ is set to 0.1. The cluster centroids are initialized with k-Means and agglomerative clustering algorithms.

The parameters are maintained constant in all the experiments.

C. Baseline

Four state-of-the-art approaches served as baselines.

1) Deep Embedded Clustering (DEC) [10]: relies on AutoEncoders as initialization method and uses k-Means for clustering. First, the method pre-trains the model using an input reconstruction loss function (non-clustering loss). Second, a clustering algorithm initializes the clustering centers, without considering the decoder. Then, the model is optimized using the Cluster Assignment Hardening (CAH) loss, and the clustering centers are updated at each iteration by minimizing the Kullback-Leibler (KL) divergence as a loss function.

2) Improved Deep Embedded Clustering (IDEC) [11]: is a modified version of DEC that preserves the decoder layer, which is used for clustering loss. The reconstruction loss of AutoEncoders is added to the objective and optimized simultaneously with clustering loss, preserving the local data structure.

3) Deep Convolutional Embedded Clustering (DCEC) [12]: uses the IDEC algorithm by incorporating a Convolutional AutoEncoder (CAE). Due to its spatial mapping relationships, CAE is able to learn deeper and spatial embedded features for clustering.

4) Deep Sensory Clustering (DSC) [6]: is an alternative to IDEC. It is based on a recurrent AutoEncoder with two different decoders: one for input sequence reconstruction and one for anticipation sequence reconstruction. By using two decoders the AutoEncoder is forced to learn highly discriminative embedded features that are used to initialize the hidden state of the decoders.

D. Datasets

The proposed architecture and baselines were evaluated on three datasets.

1) Human Activity Recognition Using Smartphones Dataset (UCI HAR) [15]: includes 3-axial linear acceleration, 3-axial angular velocity, and gyroscope sensor data. The signals were recorded with a Samsung Galaxy S II smartphone at a constant rate of 50Hz, and the activities were performed by 30 volunteers. Each subject performed 6 activities. All the participants were wearing a smartphone on the waist during the experiment execution.

2) Skoda Mini Checkpoint Activity Recognition Dataset (Skoda) [16]: includes 20 sensors (3-axial acceleration) placed on the left and right upper and lower arm. The signals were recorded at a sampling rate of 98Hz, and the activities were performed by 1 volunteer who executed 19 times each activity. The dataset contains 10 activities

3) mHealth Dataset (MHEALTH) [17]: includes 3-axial acceleration from the chest sensor, 2 electrocardiogram signals, 3-axial acceleration from the left-ankle sensor, 3-axial acceleration from the right-lower-arm sensor, 3-axial magnetometer from the left-ankle sensor, 3-axial magnetometer from the right-lower-arm sensor, 3-axial gyroscope from the left-ankle sensor, 3-axial gyroscope from the right-lower-arm sensor. The recorded activities are 12 and are performed by 10 volunteers. All sensors were recorded at a sampling rate of 50Hz.

Datasets were initially rescaled using per-channel normalization. Then, the signals have been divided in windows with an overlap between subsequent segments of 50%. For UCI HAR and MHEALTH, a window of $2.56s$ was used, while for Skoda, a window of $2s$. Furthermore, in MHEALTH, the data relative to electrocardiogram signals have been removed as the other datasets had no data of this type.

After the pre-processing phase, the first 50% of sensory measurements in each sample represents the *input sequences*. Consequently, the *temporally inverted sequence* of the input is used as the target sequence for the reconstruction task. The *remaining sensory measurements* are considered as the target sequence for future prediction.

E. Evaluation Metrics

All clustering methods are evaluated by *Clustering Accuracy (ACC)* [25] and *Normalized Mutual Information (NMI)* [26], widely used in unsupervised learning scenarios. Both metrics expect values in the range [0, 1], with 1 being the perfect clustering and 0 being the worst.

The Clustering Accuracy (ACC) takes a cluster assignment and a ground truth assignment and then finds the best matching between them. We used the *Hungarian algorithm* [27] to compute the best mapping (see Equation 18).

$$ACC = max_n \frac{\sum_{i=1}^{n} 1\{l_i = m(c_i)\}}{n} \quad (18)$$

where l_i is the true label, c_i the cluster assignment, and m ranges over all possible one-to-one mappings between clusters and labels.

The Normalized Mutual Information (NMI), between ground-truth labels and the labels obtained by clustering, determines the class labels' entropy reduction, assuming the cluster labels are known. It is used for determining the quality of the clustering and is defined in Equation 19.

$$NMI(y, c) = \frac{2I(y, c)}{H(y) + H(c)} \quad (19)$$

where y is the true label, c is the obtained cluster label, $I(.,.)$ is the mutual information and $H(.)$ is the entropy.

TABLE I: Clustering performances in terms of NMI and ACC on the three datasets achieved by DISC and state-of-the-art methods. The first group of rows shows the performance of traditional clustering techniques applied on raw data. The second group of rows shows the results achieved by DISC and state-of-the-art AutoEncoders combined with traditional clustering techniques. The last group of rows shows the comparison between DISC and the end-to-end deep clustering methods.

| | UCI HAR | | | | Skoda | | | | MHEALTH | | | | Average | | | |
| | Train | | Test | | Train | | Test | | Train | | Test | | Train | | Test | |
	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC
Traditional Clustering on Input Data Space																
k-Means	49.36	45.04	45.90	40.69	38.54	39.18	36.93	38.59	41.85	43.95	42.14	44.41	43.25	42.72	41.66	41.23
AC-Average	1.18	19.16	1.8	18.29	1.51	13.88	18.38	14.37	12.12	10.35	9.63	10.19	4.93	14.46	9.94	14.28
AC-Complete	3.45	19.56	19.22	31.69	33.29	31.32	31.86	28.07	23.49	42.10	30.02	13.74	20.07	30.99	27.03	24.50
AC-Ward	40.73	42.26	47.71	43.26	41.79	41.35	36.99	33.19	48.07	48.26	48.54	48.60	43.53	43.96	44.41	41.68
Traditional Clustering on proposed Recurrent AutoEncoding Space																
DSC k-Means	51.93	60.19	45.49	55.62	53.75	47.56	50.64	42.62	54.86	43.96	55.75	48.24	53.51	50.57	50.62	48,82
DSC AC-Average	45.18	37.57	46.41	34.61	18.88	16.96	38.59	30.22	34.54	20.47	47.01	29.26	32.86	25.00	44.00	31.36
DSC AC-Complete	40.66	40.03	40.81	43.67	32.55	32.47	41.57	35.93	42.23	35.05	44.42	36.51	38.48	35.85	42.26	38.70
DSC AC-Ward	75.27	74.78	52.83	60.33	55.81	51.51	54.41	45.96	61.07	48.91	57.04	46.28	64.05	58.40	54.76	50.85
(ours) DISC k-Means	54.84	55.27	48.27	54.62	50.65	47.83	47.87	43.10	65.14	55.27	64.88	54.04	56.88	52.76	53.67	50.59
(ours) DISC AC-Average	43.82	39.17	41.26	35.12	29.74	28.82	29.63	24.12	30.80	13.69	31.24	13.16	34.79	27.23	34.04	24.13
(ours) DISC AC-Complete	37.77	44.12	38.91	40.62	35.48	34.62	44.13	37.31	46.58	36.22	50.21	30.24	39.94	38.32	44.41	36.06
(ours) DISC AC-Ward	64.21	63.14	51.75	60.77	52.85	47.92	54.10	43.45	68.85	58.31	70.98	59.52	61.97	56.46	58.94	54.58
End-to-End Deep Clustering																
DEC [10]	55.57	49.93	55.20	49.10	46.86	41.02	46.58	41.08	50.52	43.71	51.68	45.11	50.98	44.89	51.15	45.10
IDEC [11]	55.76	51.78	54.43	51.02	49.65	46.70	48.93	40.56	51.28	42.59	52.34	44.76	52.23	47.02	51.90	45.45
DCEC [12]	52.77	48.39	53.12	48.88	42.90	39.56	45.51	40.24	44.15	23.51	45.13	24.29	46.61	40.15	47.92	38.47
DSC (k-Means) [6]	64.75	64.54	61.58	61.28	56.91	50.97	57.01	50.28	62.65	57.19	63.06	56.85	61.43	57.56	60.55	56,13
DSC (Ward) [6]	76.43	78.79	71.25	75.41	**56.97**	52.90	**59.06**	53.48	59.42	51.57	60.91	53.33	64.27	61.08	63.74	60.74
(ours) DISC (k-Means)	**85.72**	**92.08**	80.81	**87.99**	52.93	54.45	52.93	**54.45**	**68.15**	53.96	69.38	54.83	68.93	66.83	67.71	**65.76**
(ours) DISC (Ward)	85.47	81.86	**81.91**	79.40	54.91	**54.57**	55.93	53.45	63.23	50.74	61.29	48.82	67.87	62.39	66.38	60.56

IV. EXPERIMENTS RESULTS

Table I shows the clustering performance in terms of NMI and ACC on the UCI HAR, Skoda, and MHEALTH datasets achieved by the proposed and state-of-the-art methods.

We used as traditional clustering methods k-Means [28] and Agglomerative Clustering (AC) [29], with three different linkage types (i.e., Average, Complete, and Ward).

The first group of rows shows the performance of traditional clustering techniques applied on the raw data. The second group of rows shows the results achieved by our architecture DISC and state-of-the-art- AutoEncoders combined with traditional clustering techniques. The last group of rows shows the comparison of DISC method with the state of the art. Boldface font stands for best results.

Overall the results confirm the effectiveness of deep clustering approaches against traditional clustering approaches on input space data and AutoEncoding space. On average, our proposal performs better with respect to the state-of-the-art.

DSC (Ward) performs slightly better (on average of about 3%) than our proposal in terms of NMI only on the Skoda dataset. Our method overcomes the best method in the state of the art of about 10% in terms of NMI and ACC on both UCI HAR and MHEALTH. Improvements are mainly related to the use of convolutional GRU instead of simply GRU.

Figure 2 shows the progression of the learned feature space from the initial setup to the final clustering-oriented embedding space for the UCI HAR dataset. The progression is achieved by firstly using the Principal Component Analysis (PCA) [30] to reduce the embedding from 256 to 50 dimensions, and then by using the t-Distributed Stochastic Neighbor Embedding (t-SNE) [21] method applied for mapping 50-dimensional data into 2-dimensional data.

The proposed architecture discovers well-defined and separated clusters of activity segments with strong correspondence to the ground-truth labels. In particular, three activities (walking, walking upstairs, and walking downstairs) are correctly delimited, while the others (sitting, standing, and lying) are well delimited but more dispersed, due to the high number of subjects in the dataset, which increase the inter-variability of activities.

V. CONCLUSIONS AND FUTURE DIRECTIONS

The paper presents DISC, a deep-learning-based clustering architecture that learns highly discriminative spatio-temporal representations with reconstruction and future prediction tasks on multi-dimensional inertial signals. The architecture includes a recurrent encoder ConvGRU, two conditional decoders GRU, and a clustering criterion to predict unlabelled human activities-related signals.

The proposed architecture has been compared with state-of-the-art approaches in both traditional and DL-based clustering approaches, demonstrating its effectiveness on three HAR datasets.

We plan to include DISC within the Continuous Learning Platform (CLP) [31] framework. CLP is a platform that semi-automatically integrates heterogeneous labeled data and provides them in a homogeneous form. Integrating DISC
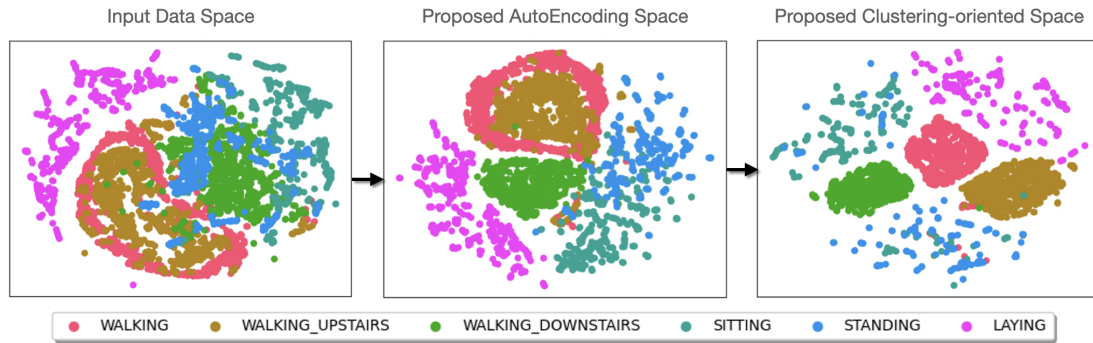
Fig. 2: Feature space progression of the UCI HAR Dataset with DISC (k-Means initialization).

into CLP would allow the dataset to be also populated with unlabeled signals that are collected with consumer devices. Moreover, DISC can be also be employed in the early stages of a personalization strategy and thus allowing the recognition of ADL to a never seen user [32].

REFERENCES

[1] S. Sun, A. A. Folarin, Y. Ranjan, Z. Rashid, P. Conde, C. Stewart, N. Cummins, F. Matcham, G. Dalla Costa, S. Simblett *et al.*, "Using smartphones and wearable devices to monitor behavioral changes during covid-19," *Journal of medical Internet research*, vol. 22, no. 9, p. e19992, 2020.

[2] D. Mukherjee, R. Mondal, P. K. Singh, R. Sarkar, and D. Bhattacharjee, "Ensemconvnet: a deep learning approach for human activity recognition using smartphone sensors for healthcare applications," *Multimedia Tools and Applications*, vol. 79, no. 41, pp. 31 663–31 690, 2020.

[3] K. Iyengar, G. K. Upadhyaya, R. Vaishya, and V. Jain, "Covid-19 and applications of smartphone technology in the current pandemic," *Diabetes & Metabolic Syndrome: Clinical Research & Reviews*, vol. 14, no. 5, pp. 733–737, 2020.

[4] A. Ferrari, D. Micucci, M. Mobilio, and P. Napoletano, "Trends in human activity recognition using smartphones," *Journal of Reliable Intelligent Environments*, vol. 7, no. 3, pp. 189–213, 2021.

[5] H. Amrani, D. Micucci, M. Mobilio, and P. Napoletano, "Homogenization of existing inertial-based datasets to support human activity recognition," *arXiv preprint arXiv:2201.07891*, 2022.

[6] A. Abedin, F. Motlagh, Q. Shi, H. Rezatofighi, and D. Ranasinghe, "Towards deep clustering of human activities from wearables," in *Proceedings of the International Symposium on Wearable Computers (ISWC)*, 2020.

[7] E. Min, X. Guo, Q. Liu, G. Zhang, J. Cui, and J. Long, "A survey of clustering with deep learning: From the perspective of network architecture," *IEEE Access*, vol. 6, pp. 39 501–39 514, 2018.

[8] G. Gan, C. Ma, and J. Wu, *Data clustering: theory, algorithms, and applications*. Society for Industrial and Applied Mathematics (SIAM), 2007.

[9] M. R. Karim, O. Beyan, A. Zappa, I. G. Costa, D. Rebholz-Schuhmann, M. Cochez, and S. Decker, "Deep learning-based clustering approaches for bioinformatics," *Briefings in Bioinformatics*, vol. 22, no. 1, pp. 393–415, 2020.

[10] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2016.

[11] X. Guo, L. Gao, X. Liu, and J. Yin, "Improved deep embedded clustering with local structure preservation," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2017.

[12] X. Guo, X. Liu, E. Zhu, and J. Yin, "Deep clustering with convolutional autoencoders," in *Proceedings of the International Conference On Neural Information Processing (ICONIP)*, 2017.

[13] A. Obeid, I. M. Elfadel, and N. Werghi, "Unsupervised land-cover segmentation using accelerated balanced deep embedded clustering," *IEEE Geoscience and Remote Sensing Letters*, vol. 19, pp. 1–5, 2021.

[14] N. Ballas, L. Yao, C. Pal, and A. Courville, "Delving deeper into convolutional networks for learning video representations," *arXiv preprint arXiv:1511.06432*, 2015.

[15] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones." in *In Proceedings of the European Symposium on Artificial Neural Networks (ESANN)*, 2013.

[16] T. Stiefmeier, D. Roggen, G. Ogris, P. Lukowicz, and G. Tröster, "Wearable activity tracking in car manufacturing," *IEEE Pervasive Computing*, vol. 7, no. 2, pp. 42–50, 2008.

[17] 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 *International Workshop on Ambient Assisted living (IWAAL)*, 2014.

[18] A. Graves, N. Jaitly, and A.-r. Mohamed, "Hybrid speech recognition with deep bidirectional LSTM," in *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2013.

[19] A. Murad and J.-Y. Pyun, "Deep recurrent neural networks for human activity recognition," *Sensors*, vol. 17, no. 11, p. 2556, 2017.

[20] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[21] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of machine learning research*, vol. 9, no. 11, pp. 2579–2605, 2008.

[22] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[24] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proccedings of the International Conference on Machine Learning (ICML)*, 2015.

[25] A. Strehl and J. Ghosh, "Cluster ensembles—a knowledge reuse framework for combining multiple partitions," *Journal of Machine Learning Research*, vol. 3, pp. 583–617, 2002.

[26] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *The Journal of Machine Learning Research*, vol. 11, pp. 2837–2854, 2010.

[27] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[28] G. Hamerly and C. Elkan, "Learning the k in k-means," *Advances in neural information processing systems*, vol. 16, pp. 281–288, 2003.

[29] D. Müllner, "Modern hierarchical, agglomerative clustering algorithms," *arXiv preprint arXiv:1109.2378*, 2011.

[30] I. Jolliffe, "Principal component analysis," *Encyclopedia of statistics in behavioral science*, 2005.

[31] A. Ferrari, D. Micucci, M. Mobilio, and P. Napoletano, "A framework for long-term data collection to support automatic human activity recognition," in *Proceedings of the Workshop on Reliable Intelligent Environment (WoRIE)*, 2019.

[32] H. Amrani, D. Micucci, and P. Napoletano, "Personalized models in human activity recognition using deep learning," in *Proceedings of International Conference on Pattern Recognition (ICPR)*, 2021.