Unsupervised Human Action Recognition with Skeletal Graph Laplacian and Self-Supervised Viewpoints Invariance

Giancarlo Paoletti[1]
giancarlo.paoletti@iit.it
Jacopo Cavazza[1]
jacopo.cavazza@iit.it
Cigdem Beyan[1,2]
cigdem.beyan@unitn.it
Alessio Del Bue[1]
alessio.delbue@iit.it

[1] Pattern Analysis and Computer Vision (PAVIS)
Istituto Italiano di Tecnologia (IIT)
Genova, Italy

[2] Department of Information Engineering and Computer Science
University of Trento
Trento, Italy

Abstract

This paper presents a novel end-to-end method for the problem of skeleton-based unsupervised human action recognition. We propose a new architecture with a convolutional autoencoder that uses graph Laplacian regularization to model the skeletal geometry across the temporal dynamics of actions. Our approach is robust towards viewpoint variations by including a self-supervised gradient reverse layer that ensures generalization across camera views. The proposed method is validated on NTU-60 and NTU-120 large-scale datasets in which it outperforms all prior unsupervised skeleton-based approaches on the cross-subject, cross-view, and cross-setup protocols. Although unsupervised, our learnable representation allows our method even to surpass a few supervised skeleton-based action recognition methods. The code is available in: www.github.com/IIT-PAVIS/UHAR_Skeletal_Laplacian

1 Introduction

Human Action Recognition (HAR) is ubiquitous across several computer vision applications, ranging from smart video surveillance, human-robot interaction, remote healthcare monitoring, and smart homes, to name a few. The recent success in skeleton-based HAR, particularly by adopting deep learning methodologies, primarily relies on the supervised learning paradigm [9, 31, 34]. However, data annotation is expensive, time-consuming, and prone to human errors [20]. As a (recent) alternative, unsupervised approaches [7, 9, 12, 18, 22, 28, 30, 35] are continuously reducing the performance gap with the fully supervised counterpart while dismissing the strong reliance over annotated data.

This paper tackles the unsupervised HAR (U-HAR) problem as formalized *e.g.* in [28] and illustrated in Fig. 1. Using unannotated 3D skeleton sequences, we learn a feature representation, which is then fed to an action recognition classifier (*e.g.*, 1-nearest neighbor) to validate the method performance as defined in standard evaluation protocols [7, 9, 12, 18, 22, 28, 30, 35]. We propose a novel unsupervised method for U-HAR that learns action representations through a ***convolutional (residual) autoencoder*** (see Fig. 2). By doing

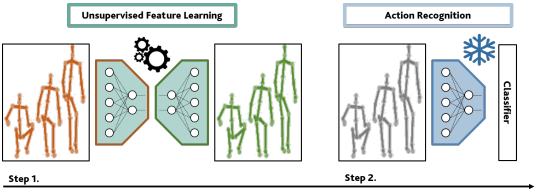


Figure 1: **Unsupervised Human Action Recognition (U-HAR) from skeleton data.** We compute features without using any supervision but by learning how to reconstruct skeleton data extracted with a generative approach. U-HAR evaluation relies on applying 1-Nearest Neighbor (1-NN) classifier or Linear Evaluation Protocol [7, 9, 12, 18, 22, 28, 30, 35].

so, we demonstrate the benefits of performing residual convolutions to jointly learn representations with spatio-temporal convolutions instead of relying on more complex and/or memory-intense architectures, which use *e.g*. contrastive learning, GANs, gated networks, or recurrent networks [7, 9, 12, 18, 22, 28, 30, 35].

To boost the performance even further, we adopt ***(graph) Laplacian regularization*** [17] to learn representations that are aware of the spatial configuration of the *skeletal geometry*. We apply this regularization in the *reconstruction* space (*i.e*., the space induced by the last layer of the decoder) to inject a "continuity pattern" while making this "approximation" smoother. This work is the first attempt where Laplacian Regularization is used within an unsupervised feature learning paradigm for action recognition.

To promote the deployment of our method in practical scenarios, we also tackle the problem of viewpoint-invariance as camera positions and orientations used to capture humans very likely differ from the setup used in the tested dataset. We improve ***viewpoint-invariance*** by, first, perturbing the original data with random rotations. Then, to increase the generalizability of the model, we enhance the unsupervised learned data representations by pairing the Laplacian-regularized reconstruction loss with a regressor head. This regressor attempts to learn the parameters (rotation angles) of the random rotations we applied. Using adversarial training in the form of a gradient reversal layer [6], we learn a feature representation that can fool this regressor, being, thus, not influenced by the rotational perturbation. This is a proxy for rotational invariance that we achieve with a different (and more effective - see Table 2 and Section 4.2) method than the Siamese network proposed in [18] (only attempting to align rotated with non-rotated data). It is important to notice that we do not achieve invariance towards some annotated features of the data, but we directly synthesize the random rotations, generated from the data itself: we thus leveraging on the concept of ***self-supervision***.

To validate our method, experiments were realized on two large-scale skeletal action datasets: NTU-60 (cross-subject and cross-view) [24] and NTU-120 (cross-subject and cross-setup) [15]. Ablation studies are performed to dissect the impact of our autoencoder, the skeletal graph Laplacian, and our adaptation of gradient reversing to U-HAR. The *end-to-end* approach we proposed outperforms prior unsupervised skeleton-based methods for U-HAR. It also favorably scores *w.r.t*. state-of-the-art supervised methods, even outperforming a few of them (see Fig. 4).

2 Related Work

Unsupervised skeleton-based HAR. Encoder-decoder recurrent architectures are often used to solve HAR problems [9, 12, 22, 28, 35]. Zheng *et al*. [35] introduce LongT GAN, based on GRUs that learns how to represent skeletal body poses in time, with an adversarial loss

supporting an auxiliary inpainting task. MS^2L [12] is also based on GRUs and benefits from contrastive learning, motion prediction, and jigsaw puzzle recognition. In addition, Kundu *et al*. [9] include a GAN-based encoder in their recurrent architecture (EnGAN). *PCRP* [30] builds upon a vanilla autoencoder trained to reconstruct the skeletal data using expectation maximization with learnable class prototypes. Su *et al*. [28] present the Predict & Cluster (P&C) method based on encoder-decoder RNN. AS-CAL [22] combines contrastive learning with momentum LSTM where the similarity between augmented instances and the input skeleton sequence is contrasted, and then a momentum-based LSTM encodes the long-term actions. SeBiReNet [18] uses a Siamese denoising autoencoder is used with feature disentanglement, showing good performance across pose denoising and unsupervised cross-view HAR. Unlike related works, our autoencoder is built on residual convolutions, showing the benefits of using simpler but superior architecture *w.r.t*. methods adapting gated or recurrent units, contrastive learning, and GANs. Recently, Li *et al*. [13] processed the joint, motion, and bone information altogether instead of using skeleton data supplied by the datasets. Using these three modalities within a contrastive learning schema, this method improved the U-HAR results of NTU-60 cross-subject and cross-view (77.8% and 83.4% respectively), and NTU-120 cross-subject and cross-setup (67.9% and 66.7%, respectively).

Laplacian Regularization. Belkin *et al*. [17] propose to regularize a model using the implicit geometry of the feature space, regardless of the distribution of their labels, by using the Laplacian of the graph built over the cross-similarity of examples. A similar approach was pursued by a recent end-to-end trainable approach for image denoising [19]. We follow a different approach by applying Laplacian regularization in space while our autoencoder learns to reconstruct input skeletal data (*i.e*., *reconstruction* space). In this way, our goal is to inject the information of skeletal geometry into our model. Different from *supervised* HAR methods (*e.g*., [14, 32]) that directly exploit the "raw" adjacency matrix to encode skeletal connectivity, we take advantage of a more powerful mathematical tool, the graph Laplacian, since it better capitalizes from the skeletal geometry. This differs from prior works, *e.g*. [28, 35] relying on Mean-Squared Error (MSE)-based action reconstruction only.

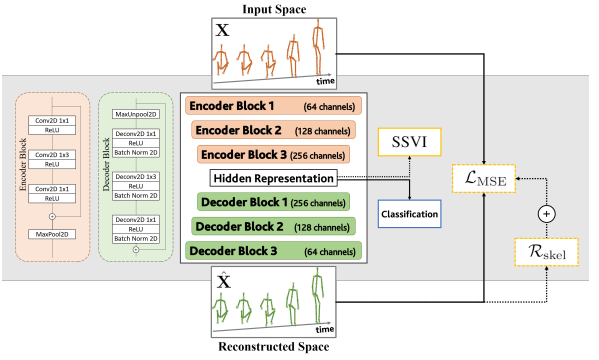
Gradient Reversing. Originally proposed for domain adaptation, the gradient reversal layer (GRL) [6] is arguably useful to achieve a better generalization: *e.g*., classify actions performed by multiple subjects [36]. Differently, we propose to endow a non-discriminative architecture (an autoencoder) for viewpoint-invariance. We perform this by first synthesizing auxiliary rotations of the skeletal joints to simulate different viewpoints. Then, by achieving the invariance across viewpoints by a GRL layer that is fooling a predictor attempting to infer the viewpoint from the hidden representation of our autoencoder. Li *et al*. [10] adapts GRL to obtain view-invariant action representations. However, that work differs from ours by (a) relying on RGB-D data and, more importantly, (b) using the annotated viewpoints of the datasets as source and target domains and learn how to distinguish them by classification.

3 Proposed Method

We present our unsupervised approach by introducing the proposed convolutional autoencoder (Section 3.1) and the Laplacian regularization (Section 3.2). Following that, we discuss the self-supervised viewpoint invariance module (SSVI; Section 3.3).

3.1 Convolutional Autoencoder

The proposed Convolutional Autoencoder (**AE**) input is a set of 3D human body joints in time extracted from a video sequence with one or more subjects performing an unlabelled action. Let $\mathbf{X}$ denote an input sequence of body joints represented as a $d \times m \times t$ tensor,



Algorithm 1 Training our proposed approach

Randomly initialize $\mathbf{E}_\varphi$, $\mathbf{D}_\theta$ and the SSVI module.
Compute the skeletal graph Laplacian $\mathbf{L}$.
while not converged **do**
  Sample a mini-batch of data $\mathcal{B}$.
  Do a forward pass through $\mathbf{E}_\varphi$ and $\mathbf{D}_\theta$.
  Update $\mathbf{E}_\varphi$, $\mathbf{D}_\theta$ using the MSE loss as in Eq (1).
  ▷ (OPTIONAL SKELETAL LAPLACIAN REGULARIZATION)
  Update $\mathbf{E}_\varphi$, $\mathbf{D}_\theta$ using the $\mathcal{R}_{\text{skel}}$ loss as in Eq (2).
  ▷ (OPTIONAL VIEWPOINTS INVARIANCE)
  Randomly sample α, β, γ in $[0, 2\pi]$
  Rotate all data in $\mathcal{B} \longrightarrow \mathcal{B}^{(\alpha,\beta,\gamma)}$
  Do a forward pass through $\mathbf{E}_\varphi$
  Update $\mathbf{E}_\varphi$ using the SSVI module fed by $\mathcal{B}^{(\alpha,\beta,\gamma)}$.
end while

Figure 2: **Our proposed method (Left):** we exploit a convolutional autoencoder (**AE**) trained with $\mathcal{L}_{\text{MSE}}$ (Eq. (1)). In the reconstruction space, we perform a *Skeletal Laplacian Regularization* (**L**; Sec. 3.2, Eq. (2)), enriching the learned (hidden) feature representations with the skeletal geometry information. We also include a *self-supervised viewpoint-invariance* (**SSVI** module, Sec. 3.3), which adapts a gradient reversal layer [6] to achieve robustness towards different viewpoints. Our convolutional encoder and deconvolutional decoder blocks both exploit residual connections while batch normalization is exclusive for the decoder. **Right:** Pseudo-code of the training process.

containing the x, y, z coordinates ($d = 3$), the number of joints ($m = 25$ on NTU-60 [24] and NTU-120 [15]) and the number of timestamps t[1]. We aim at obtaining *unsupervised feature representations* by learning an autoencoder that reconstructs the input data $\mathbf{X}$ using a Mean-Squared Error (MSE) loss:

$$\mathcal{L}_{\text{MSE}} = \tfrac{1}{2}\mathbb{E}_{\mathbf{X}\sim\mathcal{B}}\left[\|\mathbf{X} - \hat{\mathbf{X}}\|_F^2\right], \tag{1}$$

where $\|\cdot\|_F$ denotes the Frobenius norm, *i.e.*, the Euclidean norm of the vector obtained after flattening the tensor. The MSE loss in Eq. (1) is minimized by using gradient descent (Adam optimizer) over mini-batches $\mathcal{B}$. The reconstructed data are defined as $\hat{\mathbf{X}} = \mathbf{D}_\theta \circ \mathbf{E}_\varphi(\mathbf{X})$ and computed using an encoder-decoder architecture, where φ denotes the learnable parameters of the encoder $\mathbf{E}$ and θ are the analogous parameters for the decoder $\mathbf{D}$. The complete architecture of our convolutional autoencoder is detailed in Fig. 2. We concatenate different residual blocks to build the autoencoder: 3 for the encoder and 3 for the decoder. At the end of the encoder, a FC layer represents the latent space $\mathbf{z}$ of size 2048. The size of $\mathbf{z}$ was determined by testing various numerical combinations, *e.g.*, 32, 128, 512. For our convolutional autoencoder, 2048 results in the best performances (up to +10% in NTU-60 and +23% in NTU-120) out of all combinations. Thus, this value was fixed in all experiments.

Residual blocks of convolutions. Our **AE** architecture stacks different fully-residual blocks for both encoder and decoder, whereas each block is made of convolutions capable to jointly learning spatial representations of skeletal data in time, treating each skeletal data $\mathbf{X}$ as 2D convolutions. Convolutions with fixed size kernels (either 1×1 or 1×3), applied inside $\mathbf{E}$ and $\mathbf{D}$, are capable to capture spatial and temporal relationships of data along tensor rows for the former and along tensor columns for the latter. Hence it is called *convolutions-in-time*. In detail, within the encoder blocks, the residual layer is made of a series of three *2D-convolutional* layers (each with *ReLU* activations) stacked together. At the same time, decoder blocks share a similar structure but using instead *2D-deconvolutional* layers with the addition of *2D-BatchNorm* applied after each *ReLU* activation. To ensure the bottleneck

[1] To be comparable with prior art, we cast each skeleton sequence to a fixed temporal length [28].

structure of the convolutional autoencoder, a *MaxPool* layer is applied at the end of each encoder block, whereas a *MaxUnpool* layer is applied at the beginning of each decoder block (see Fig. 2).

3.2 Skeletal Laplacian Regularization

The graph Laplacian is an established tool to analyze weighted undirected graphs. It builds upon the graph adjacency matrix $\mathbf{W}$, whose entries W_{ij} are defined such that $W_{ij} = 1$ if and only if the nodes i and j are connected through an edge. The (un-normalized) graph Laplacian $\mathbf{L}$ is easily computable from $\mathbf{W}$ as $\mathbf{L} = \mathbf{D} - \mathbf{W}$, where $\mathbf{D}$ is the degree matrix (obtained as the diagonal matrix where its (i,i)-th element is $D_{ii} = \sum_j W_{ij}$) [5]. The Laplacian regularizer $\mathcal{R}(\mathbf{z}) = \sum_{i,j} W_{ij}(z_i - z_j)^2$ can be applied to a hidden vectorial embedding $\mathbf{z}$ to learn the geometry of the feature space (where $\mathbf{z}$ belongs to) and to capitalize from these cues to solve a semi-supervised learning paradigm [17]. This is true because, thanks to the weights W_{ij}, we can prioritize the alignment between the scalar components z_i and z_j by simply putting a stronger penalty between pairs of components that must be well aligned. In our case, we attempt to do so by promoting the alignment of skeletal joints, which are connected through a bone (*e.g.*, *an edge exists if and only if joints are connected*). The results given in Supplementary Material show that such a setting is also empirically favorable compared to other ways of initializing $\mathbf{W}$. We intend this as a valid proxy for injecting the knowledge of skeletal geometry while learning our action representations. The reason why $\mathcal{R}$ is termed Laplacian regularizer lies in the fact that $\mathcal{R}(\mathbf{z}) = 2\,\mathbf{z}^\top \mathbf{L}\mathbf{z}$. That is, $\mathcal{R}(\mathbf{z})$ implements a "$\mathbf{L}$-weighted weight decay" - since $\mathcal{R}(\mathbf{z}) = \|\mathbf{Q}\mathbf{z}\|_2^2$ if we set $\mathbf{Q} = \sqrt{\mathbf{L}}$.

Unlike prior art [17, 19], we apply Laplacian regularization to the ***reconstruction space*** learned by our decoder, *i.e.*, the space where $\hat{\mathbf{X}}$ belongs to. We compute our proposed ***skeletal Laplacian regularizer*** as:

$$\mathcal{R}_{\text{skel}} = \mathbb{E}_{\mathbf{X}\sim\mathcal{B}}\left[\mathbb{E}_{t,d}\left[\hat{\mathbf{x}}^{(t,d)\top}\mathbf{L}\hat{\mathbf{x}}^{(t,d)}\right]\right], \tag{2}$$

where $\hat{\mathbf{x}}^{(t,d)}$ is the m-dimensional column vector stacking the scalar (abscissæ, ordinatæ or quotæ) coordinates along the dimension d obtained from the reconstructed sequence $\hat{\mathbf{X}}$ at time t. In Eq. (2), the regularizer $\mathcal{R}_{\text{skel}}$ is averaged over the mini-batch $\mathcal{B}$, considering the reconstructions produced by the convolutional autoencoder across coordinates and timestamps. The Laplacian regularization attempts to inject the connectivity of the skeleton to learn a feature representation, which is aware of the ***skeletal geometry***. We deem this to be a proxy of features that are aware of the fact that the representation learned, *e.g.*, from the shoulder and elbow joints, cannot be decorrelated from each other since those joints are closed in space, while there can be joints, which are more distant in space (*e.g.*, left foot vs. right hand) are allowed to be more independent.

3.3 Self-supervised Viewpoints Invariance (SSVI)

We propose to obtain a viewpoint-invariant action representation by first synthesizing multiple viewpoints of the original skeletal data. Geometrically, this operation can be easily framed as (right) multiplying $\mathbf{X}^t$, the $m \times 3$ matrix stacking the m 3D joints captured at a given timestamp t, by Ω defined as the product of Ω_x, Ω_y, and Ω_z, each corresponding to the independent three (planar) rotations performed around the x, y, z axis, respectively. Ω_x depends upon the pitch angle α, Ω_y depends upon the yaw angle β, and Ω_z depends upon the roll angle γ. By means of the so-defined Ω, we can obtain $\mathbf{Z}^t = \mathbf{X}^t\Omega$, and, hence, ***synthesize*** a rotation under a ***generated viewpoint*** by iterating the process over all timestamps t of the sequence $\mathbf{X}$ and, afterward repeating the whole procedure for all sequences $\mathbf{X}$ in the

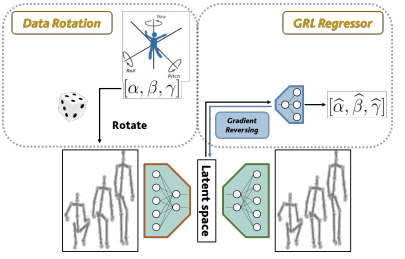


Figure 3: **Self-Supervised Viewpoints Invariance** using a regressor and a gradient reversal layer [6]. We enforce the hidden representation, learned by our encoder to be invariant across synthetic rotations that we applied to the input data $\mathbf{X}$, using the Euler's angles α, β, γ. We claim it to be the proxy to achieve viewpoints invariance and generalize across random rotations (parametrized by Euler's angles α, β, γ).

mini-batch $\mathcal{B}$, we generate transformed sequences $\mathbf{Z}$. When $\mathbf{Z}$ is obtained from $\mathbf{X}$ according to this procedure, the action class referring to them remains unaltered in its information content, while only the viewpoint had changed.

We make $\mathbf{Z}$ and $\mathbf{X}$ indistinguishable, being the latter a proxy for an improved hidden representation that our autoencoder learns from data, since in this way, the autoencoder will be robust towards different viewpoints, and we purport that this requirement is a proxy for an improved viewpoint generalization. We use a L1 norm to train a ***regressor*** that predicts the triplet $[\alpha, \beta, \gamma]$ that is used to rotate the data. We take advantage of a ***gradient reversal layer (GRL)*** [6] to flip the gradients coming from the regressor. By doing so, we make sure to actively promote invariance across synthetic rotations by explicitly optimize the learned representation to fool a regressor that is attempting to predict the $[\alpha, \beta, \gamma]$ triplet used to rotate the data of each mini-batch before every forward pass. We call this the ***self-supervised viewpoints invariance*** **(SSVI)** module, which is visualized in Fig. 3 and connected to the hidden representation of the autoencoder (see Fig. 2).

3.4 Pseudo-code & Inference

The pseudo-code of our method is given in Algorithm 1 (see Fig. 2 Right). Below, we provide details on how a trained autoencoder is used for inference. All implementation details, including learning curves, can be found in the Supplementary Material.

Inference. Standard evaluation protocols for U-HAR perform inference by keeping the learned representations frozen and training a classifier on top of them. Two alternatives are used: a *linear classifier* using the available labels of the datasets (***Linear Evaluation Protocol (LEP)***) [7, 9, 18, 22, 30, 35] and a 1-nearest neighbor predictor (1-***NN***) [28].

4 Experimental Analysis

The proposed method is validated using the two large-scale skeletal action datasets: NTU-60 [24] and NTU-120 [15]. **NTU-60** dataset contains 60 action classes performed by 40 subjects, captured with Microsoft Kinect v2 (25 joints). **NTU-120** encompasses 120 action classes of 106 subjects while there are in total 32 different setups (*e.g.*, different backgrounds or locations where the data is captured). We evaluate the proposed method on NTU-60 dataset for cross-subject (C-Subject) and cross-view (C-View) settings [24], and NTU-120 for cross-subject (C-Subject) and cross-setup (C-Setup) settings [15].

NTU-60 [24]	C-Subject ACC (%)	C-View ACC (%)
1-NN Protocol [28]		
P&C FS* [28]	50.6	76.3
P&C FW* [28]	50.7	76.1
Baseline **AE**	50.1	80.4
Our **AE**	*52.3*	*81.0*
Our **AE-L** (AE + $\mathcal{R}_{\text{skel}}$)	***54.1***	***83.1***
Linear Evaluation Protocol (LEP) [35]		
LongT GAN [35]	39.1	48.1
MS^{2}L [20]	52.5	–
PCRP [30]	53.9	63.5
VAE-PoseRNN [9]	56.4	63.8
AS-CAL [22]	58.5	64.6
MM-AE [7]	61.2	70.2
EnGAN-PoseRNN [9]	68.6	77.8
SkeletonCLR joint [13]	68.3	76.4
Baseline **AE**	68.5	84.3
Our **AE**	*69.2*	*85.1*
Our **AE-L** (AE + $\mathcal{R}_{\text{skel}}$)	***69.9***	***85.4***

NTU-120 [15]	C-Subject ACC (%)	C-Setup ACC (%)
1-NN Protocol [28]		
P&C† [28]	41.7	42.7
Baseline **AE**	40.2	44.3
Our **AE**	*41.0*	*44.5*
Our **AE-L** (AE + $\mathcal{R}_{\text{skel}}$)	***42.4***	***44.7***
Linear Evaluation Protocol (LEP) [35]		
PCRP [30]	41.7	45.1
AS-CAL [22]	48.6	49.2
Baseline **AE**	56.4	60.3
Our **AE**	*57.1*	*61.8*
Our **AE-L** (AE + $\mathcal{R}_{\text{skel}}$)	***59.1***	***62.4***

Table 1: Evaluation on NTU-60 [24] and NTU-120 [15]. Our numbers are in italic, an improved performance over prior art is underlined. The best of all results are in black. "Baseline AE" stands for the proposed AE without residual layers. *FS and FW stand for a decoder with "fixed states" and "fixed weights", respectively [28]. †Taken from PCRP [30].

4.1 Comparisons against the state-of-the-art

Herein, we discuss the improvements that our proposed approach (AE-L) brings in, that is summarized as: for 1-NN Protocol, AE-L performs +3.4% and +6.8% on NTU-60 C-Subject and C-View, respectively, and +0.7% and +2.0% on NTU-120 C-Subject and C-Setup, respectively, over [28]. For LEP, we improve prior art on NTU-60 C-Subject (+1.3%), NTU-60 C-View (+7.6%), on NTU-120 C-Subject (+10.5%) and on NTU-120 C-Setup (+13.2%). Detailed discussion is provided below.

Cross-subject evaluation protocol. We compare our AE-L against state-of-the-art methods (SOTA) using recommended training and testing splits of NTU-60 [24] and NTU-120 [15] datasets. We use only skeletal data in our experiments, *i.e.*, we discard the RGB and depth images, normalizing data as in prior works [28], and we feed our $\mathcal{R}_{\text{skel}}$-regularized autoencoder (AE-L) with *training data only* (*i.e.*, with the data of subjects in training). Then, we apply one of the two evaluations: 1-NN or LEP, as described in Section 3.4. Readers can refer to Table 1 (C-Subject columns) for the results of the analysis mentioned above.

Ablation study shows that AE-L improves the performance of AE model, demonstrating the advantages of using Laplacian regularization: +1.8% in 1-NN, +0.7% in LEP for NTU-60 C-Subject and +1.4% in 1-NN, +2% in LEP for NTU-120 C-Subject setting. In addition, the usage of Laplacian regularization grants at least a +5% performance gain over different action classes for both NTU-60 C-Subject and C-View, and NTU-120 C-Subject and C-Setup settings (the complete list of these action classes can be found in Supplementary Material). Our AE is preferable compared to the baseline AE (i.e., not using residual layers in our design) as performing +2.2% in 1-NN, +0.7% in LEP for NTU-60 C-Subject, and +0.8% in 1-NN, +0.7% in LEP for NTU-120 C-Subject, showing the contribution of using residual convolutions layers.

For NTU-60 C-Subject, the learned features of our AE-L and AE models are superior to P&C [28]: +3.5% as compared to P&C FS [28] and +3.4% as compared to P&C FW [28]. While exploiting LEP, our AE-L again performs better than the approaches based on RNNs [9, 22], performing +11.4% better than AS-CAL [22] and +8.7% than MM-AE [7]. We improve VAE-PoseRNN [9], EnGAN-PoseRNN [9] and SkeletonCLR joint [13] by

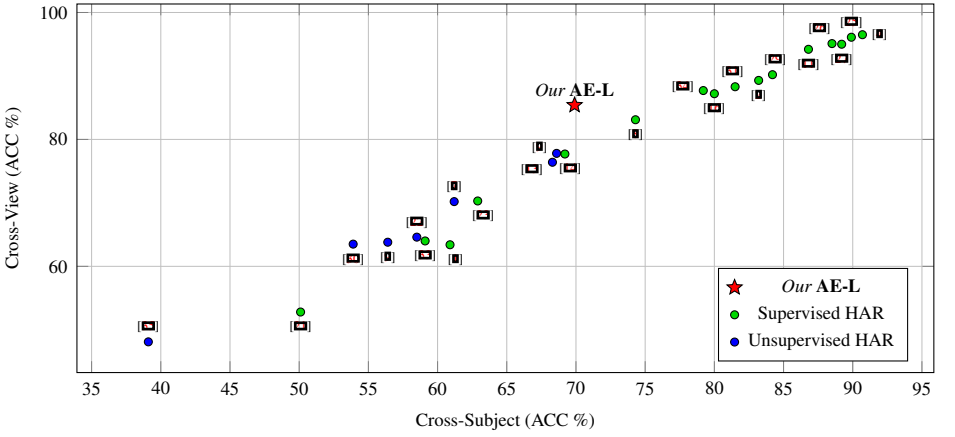


Figure 4: Comparisons between our AE-L and SOTA unsupervised and supervised skeleton-based HAR methods on NTU-60 dataset [24].

+13.5%, +1.3%, +1.6%, respectively. We also surpass MS^{2}L [12] (+17.4%). For NTU-120 C-Subject, AE-L outperforms P&C [28] (+0.7%) when 1-NN is applied, with an increase in performance *w.r.t.* both AS-CAL [22] (+10.5%) and PCRP [30] (+17.4%) in LEP.

Cross-view and cross-setup evaluation protocols. We compare our AE-L against prior methods on NTU-60 [24] C-View and NTU-120 [15] C-Setup settings (see Table 1, C-View and C-Setup columns). For NTU-60 [24] C-View, AE-L improves the performance by +6.8% and +7.0% over P&C FS [28] and P&C FW [28], respectively within the 1-NN Protocol. In the same protocol, ablation study shows that AE-L improves the performance of Baseline AE by +2.1%, and using residual layers (i.e., our AE) performs 0.6% better than not using (Baseline AE). On NTU-60 [24] C-View, with LEP, the superiority of AE-L is much visible such that it notably exceeds LongT GAN [35] (+37.3%), PCRP [30] (+21.9%), AS-CAL (+20.8%), VAE-PoseRNN (+21.6%), MM-AE (+15.2%), EnGAN-PoseRNN (+7.6%) and SkeletonCLR joint [13] (+9%). Our AE also surpasses "Baseline AE" by 0.8%, once again showing the positive contribution of residual layers. On NTU-120 [15] C-Setup, AE-L again performs better than P&C within the 1-NN Protocol (+2.0%), and in LEP, it performs better than AS-CAL and PCRP by margins of +13.2% and +17.3%, respectively. In this setting, our AE achieves better results than "Baseline AE" by +0.2% for 1-NN and +1.5% for LEP. **Confusion matrices** belonging to AE-L in testing can be found in Supplementary Material.

4.1.1 Comparisons Against Supervised Methods

We compare the performance of our AE-L with SOTA supervised skeleton-based HAR approaches on NTU-60 dataset [24]. This comparison includes kernel-based methods [1, 21] and the methods realizing feature learning [2, 4, 8, 11, 12, 16, 23, 25, 26, 27, 29, 31, 33, 34] with several different deep learning architectures, *e.g.*, RNNs, LSTMs, CNNs, and Graph Convolutional Networks (GCNs). The corresponding results are presented in Fig. 4, while an in-depth comparison is given in the Supplementary Material.

Our AE-L, although based on unsupervised learning, is able to achieve better performance than the fully supervised kernel-based methods [1, 21], with a +7.2% to +19.8% improvement in C-Subject and a +22% to +32.6% improvement in C-View setting. AE-L also outperforms several fully supervised deep architectural methods: hierarchical RNN [33] (providing an increase of 10.8% in C-Subject and up to 21.4% in C-View), spatial-temporal LSTM [14] (resulting in a boost of +0.7% in C-Subject and up to +7.7% in C-View) and

U-HAR: Transfer Across Viewpoints		# of params.	where?	NTU-60 C-View	NTU-120 C-Setup
Baseline	[28]	0.58M	input (pre-proc)	76.3%	42.7%
SeBiReNet	[18]	0.27M	input (data-aug)	79.7%	–
Our **GRAE**	(AE + SSVI)	*0.39M*	*feature space*	*81.9%*	*47.0%*
Our **GRAE-L**	(AE + $\mathcal{R}_{\text{skel}}$ + SSVI)	*0.39M*	*feature space*	***82.4%***	***48.9%***

Table 2: We compare our SSVI module, plugged into either AE and AE-L (Linear Evaluation Protocol [65], our numbers in italic) with published results of [18, 28].

part-aware LSTM [23] (achieving an improvement of +7% in C-Subject and up to +15.1% in C-View) while performing better than temporal CNN [8] (up to +2.3%) in C-View setting. These results show that our unsupervised residual convolutions with Laplacian regularization exceed even supervised GRUs, RNNs, and LSTMs (and variants) for HAR. Besides the mentioned favorable results of our AE-L, it is important to note that fully supervised techniques [2, 4, 11, 16, 25, 26, 27, 29, 31, 34] perform better than our AE-L. These methods mostly implement GCNs [4, 11, 26, 27, 29, 31], and some of them additionally adapt LSTMs [27] or a variable temporal dense block [29]. The best performing method is [4] with 90.7% and 96.5% in C-Subject and C-View, respectively.

4.2 Transfer across viewpoints for U-HAR

Since U-HAR is, by design, better tailored to real-world applications, we intended to push our approach to the limit and compete against SeBiReNet [18] to transfer across viewpoints. SeBiReNet and our GRAE-L (AE + $\mathcal{R}_{\text{skel}}$ + SSVI) leverage random rotational noise to perturb the input data with a sharp algorithmic difference. The two-stream Siamese architecture of SeBiReNet [18] is jointly fed by rotated and non-rotated data while using non-adversarial optimization to promote viewpoints invariance. Differently, we exploit gradient reversing [6] to achieve viewpoint invariance in a model which is fed by *rotated data only*, attempting to fool a regressor (one ReLU-hidden layer MLP with a sigmoid readout layer) to predicting the triplet of Euler's angles used to rotate each mini-batch (see Algorithm 1). We rely on a single stream, and as opposed to having two lightweight streams helping each other in generalizing better [18], our network is deeper (also depends upon a greater number of learnable parameters - 0.27M versus 0.39M, see Table 2) but achieves a better invariance across viewpoints. Furthermore, our approach does not benefit from auxiliary skeletal datasets as commonly happening in unsupervised domain adaptation [6] (*e.g.*, in SeBiReNet [18], a pre-training is performed on Cambridge-Imperial APE dataset, and then transfer learning is applied for NTU-60). As seen in Table 2, our GRAE (AE+SSVI) and GRAE-L (AE-$\mathcal{R}_{\text{skel}}$+SSVI) approaches score favorably against SeBiReNet [18], and GRAE-L has a +2.7% on NTU-60 C-View setting. In the same table, we also report a comparison with the baseline solution [28], applying view-invariant transformations to "clean" the data from rotations as pre-processing. Notably, despite our data being trained with more complex data to be fitted (our single-stream GRAE-L never sees non-rotated data), we still outperform this baseline by big margins (+6.1% on NTU-60 [24] C-View and +6.2% on NTU-120 [15] C-Setup).

4.3 Using synthetic data in training

This section investigates the impact of using synthetic data in training for C-View and C-Setup scenarios. Synthetic data were obtained as described in Section 3.3 (also Supplementary Material), and it was fixed for all experiments in Table 3. The models are trained with a) real data only, b) real + synthetic data, c) synthetic data only. Real data refers to pre-processed data, so-called clean data in Section 4.2, which is already aligned to the same viewpoint. Recalling that SSVI-based experiments rely only on synthetic data, whose

		AE	AE-L	GRAE (AE + SSVI)	GRAE-L (AE-L + SSVI)
NTU-60 [24] C-View	Real data (pre-processed)	85.1	85.4	~	~
	Real + Synthetic data	80.4	80.6	~	~
	Synthetic data	80.1	81.3	**81.9**	**82.4**
NTU-120 [15] C-Setup	Real data (pre-processed)	61.8	62.4	~	~
	Real + Synthetic data	45.7	45.2	~	~
	Synthetic data	46.1	46.4	**47.0**	**48.9**

Table 3: Performances (accuracy) of the proposed methods using the real and/or synthetic data in training. *Notice that methods with SSVI rely only on synthetic data.*

	NTU-60 [24]		NTU-120 [15]	
	C-Subject	C-View	C-Subject	C-Setup
Our AE (Unsupervised)	52.3	81.0	41.0	44.5
Our AE End-to-end (Supervised)	69.8	83.7	57.1	59.6
Our AE Fine-tuning (Supervised)	70.5	83.8	57.5	61.1

Table 4: Performances of our AE (accuracy) in different learning schemes.

amount is as much as the real training data, the training set size of real + synthetic experiments is twice of real only and synthetic only. Synthetic data includes rotational perturbations of *not pre-processed* real data. Thus, experiments only with synthetic data and real + synthetic data result in performance degradation for all models. Experiments with real data perform the best out of all, but it is important to notice that *the applied pre-processing is mostly not applicable in real-world applications as the viewpoints might not be known.* When the amount of synthetic data in real + synthetic setting is decreased, performance increases, e.g., AE performs 83.2% and 46.8%, AE-L performs 84.3% and 47.4% on NTU60, and NTU120 with "real + (20%)synthetic data". In this case, SSVI-based models perform better than AE and AE-L (both synthetic & real + synthetic) for all cases, showing that they can handle *viewpoint perturbations* in a better way.

4.4 Fine-tuning and end-to-end supervised training

The performances of *Our AE* with the fine-tuning protocol [13] and end-to-end supervised training are reported in Table 4. **Fine-tuning protocol** refers to first end-to-end pre-training of our AE in an unsupervised way and then appending a linear classifier to the encoder of AE, which is trained for HAR using the action labels. It was applied for 100 epochs with learning rate 0.001. **End-to-end training** refers to supervised training of our AE from scratch using the action labels of training data. It was applied for 100 epochs with learning rate 0.001. While fine-tuning performs the best out of all, fine-tuning and supervised HAR results are always better than "Our AE Unsupervised" (as expected) by +3-18% for NTU-60 [24] and +15-17% for NTU-120 [15]. As we train the Laplacian regularizer on the reconstructed skeleton from the *decoder*, and the experiments presented herein are regarding applying a linear classifier appended to the *encoder*, the results of *Our AE-L* are the same as *Our AE*.

5 Conclusion

We have introduced a novel unsupervised feature learning method that results in effective feature representations of actions from the input 3D skeleton sequences. Our method is based on convolutional autoencoders (AE) and adapting Laplacian Regularization (L) to capturing the pose geometry in time. Our AE-L is validated on large-scale HAR benchmarks where it exceeds all of SOTA skeleton-based U-HAR methods for cross-subject, cross-view, and cross-setup settings. This proves that our AE-L is able to learn more distinctive action features compared to prior art. We also upgrade AE-L with gradient reversing (GRAE-L) to provide better invariance to camera viewpoint changes compared to a direct competitor [18]. As future work, we will focus on enforcing the spatio-temporal connectivity through regularization over time and also concentrate on the real-time deployment of our AE-L.

References

[1] Jacopo Cavazza, Pietro Morerio, and Vittorio Murino. Scalable and compact 3D action recognition with approximated rbf kernel machines. *Pattern Recognition*, 93:25–35, 2019.

[2] Chao Li, Qiaoyong Zhong, Di Xie, and Shiliang Pu. Skeleton-based action recognition with convolutional neural networks. In *2017 IEEE International Conference on Multimedia Expo Workshops (ICMEW)*, pages 597–600, 2017. doi: 10.1109/ICMEW.2017.8026285.

[3] Ke Cheng, Yifan Zhang, Xiangyu He, Weihan Chen, Jian Cheng, and Hanqing Lu. Skeleton-based action recognition with shift graph convolutional network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[4] Ke Cheng, Yifan Zhang, Xiangyu He, Weihan Chen, Jian Cheng, and Hanqing Lu. Skeleton-based action recognition with shift graph convolutional network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 183–192, 2020.

[5] Narsingh Deo. *Graph theory with applications to engineering and computer science*. Courier Dover Publications, 1974.

[6] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *The International Conference on Machine Learning (ICML)*, 2015.

[7] Daniel Holden, Jun Saito, Taku Komura, and Thomas Joyce. Learning motion manifolds with convolutional autoencoders. In *SIGGRAPH Asia 2015 Technical Briefs*, 2015.

[8] T. S. Kim and A. Reiter. Interpretable 3d human action analysis with temporal convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1623–1631, 2017. doi: 10.1109/CVPRW.2017.207.

[9] Jogendra Nath Kundu, Maharshi Gor, Phani Krishna Uppala, and R Venkatesh Babu. Unsupervised feature learning of human actions as trajectories in pose embedding manifold. In *IEEE Winter Conference on Applications of Computer Vision (WACV).*, 2018.

[10] Junnan Li, Yongkang Wong, Qi Zhao, and Mohan S Kankanhalli. Unsupervised learning of view-invariant action representations. *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[11] M. Li, S. Chen, X. Chen, Y. Zhang, Y. Wang, and Q. Tian. Actional-structural graph convolutional networks for skeleton-based action recognition. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3590–3598, 2019. doi: 10.1109/CVPR.2019.00371.

[12] Lilang Lin, Sijie Song, Wenhan Yang, and Jiaying Liu. Ms2l: Multi-task self-supervised learning for skeleton based action recognition. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 2490–2498, 2020.

[13] Li Linguo, Wang Minsi, Ni Bingbing, Wang Hang, Yang Jiancheng, and Zhang Wenjun. 3d human action representation learning via cross-view consistency pursuit. In *CVPR*, 2021.

[14] Jun Liu, Amir Shahroudy, Dong Xu, and Gang Wang. Spatio-temporal lstm with trust gates for 3d human action recognition. *arXiv preprint arXiv:1607.07043*, 2016.

[15] Jun Liu, Amir Shahroudy, Mauricio Perez, Gang Wang, Ling-Yu Duan, and Alex C. Kot. Ntu rgb+d 120: A large-scale benchmark for 3d human activity understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. doi: 10.1109/TPAMI.2019.2916873.

[16] Mengyuan Liu, Hong Liu, and Chen Chen. Enhanced skeleton visualization for view invariant human action recognition. *Pattern Recogn.*, 68:346–362, August 2017. doi: 10.1016/j.patcog.2017.02.030.

[17] Vikas Sindhwani Mikhail Belkin, Partha Niyogi. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *International Journal of Machine Learning Research (JMLR)*, 7(11), 2006.

[18] Qiang Nie, Ziwei Liu, and Yunhui Liu. Unsupervised human 3d pose representation with viewpoint and pose disentanglement. In *Springer European Conference on Computer Vision (ECCV)*, 2020.

[19] J. Pang and G. Cheung. Graph laplacian regularization for image denoising: Analysis in the continuous domain. *IEEE Transactions on Image Processing*, 26(4):1770–1785, 2017.

[20] Giancarlo Paoletti, Jacopo Cavazza, Cigdem Beyan, and Alessio Del Bue. Subspace clustering for action recognition with covariance representations and temporal pruning. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, 2020.

[21] H. Rahmani, A. Mahmood, D. Huynh, and A. Mian. Histogram of oriented principal components for cross-view action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(12):2430–2443, 2016. doi: 10.1109/TPAMI.2016.2533389.

[22] Haocong Rao, Shihao Xu, Xiping Hu, Jun Cheng, and Bin Hu. Augmented skeleton based contrastive action learning with momentum lstm for unsupervised action recognition, 2020.

[23] A. Shahroudy, J. Liu, T. Ng, and G. Wang. Ntu rgb+d: A large scale dataset for 3d human activity analysis. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1010–1019, 2016. doi: 10.1109/CVPR.2016.115.

[24] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+d: A large scale dataset for 3d human activity analysis. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2016.

[25] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Skeleton-based action recognition with directed graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[26] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *CVPR*, 2019.

[27] Chenyang Si, Wentao Chen, Wei Wang, Liang Wang, and Tieniu Tan. An attention enhanced graph convolutional lstm network for skeleton-based action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[28] K. Su, X. Liu, and E. Shlizerman. Predict & cluster: Unsupervised skeleton based action recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9628–9637, 2020. doi: 10.1109/CVPR42600.2020.00965.

[29] Yu-Hui Wen, Lin Gao, Hongbo Fu, Fang-Lue Zhang, and Shihong Xia. Graph cnns with motif and variable temporal block for skeleton-based action recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):8989–8996, Jul. 2019. doi: 10.1609/aaai.v33i01.33018989. URL https://ojs.aaai.org/index.php/AAAI/article/view/4929.

[30] Shihao Xu, Haocong Rao, Xiping Hu, and Bin Hu. Prototypical contrast and reverse prediction: Unsupervised skeleton based action recognition. In *arXiv preprint 2011.07236*, 2020.

[31] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. *AAAI*, pages 7444–7452, 2018.

[32] Dong Yang, Monica Mengqi Li, Hong Fu, Jicong Fan, and Howard Leung. Centrality graph convolutional networks for skeleton-based action recognition. In *European Conference on Computer Vision (ECCV)*, 2020.

[33] Yong Du, W. Wang, and L. Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1110–1118, 2015. doi: 10.1109/CVPR.2015.7298714.

[34] Pengfei Zhang, Cuiling Lan, Junliang Xing, Wenjun Zeng, Jianru Xue, and Nanning Zheng. View adaptive recurrent neural networks for high performance human action recognition from skeleton data. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[35] Nenggan Zheng, Jun Wen, Risheng Liu, Liangqu Long, Jianhua Dai, and Zhefeng Gong. Unsupervised representation learning with long-term dynamics for skeleton based action recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

[36] Andrea Zunino, Jacopo Cavazza, Riccardo Volpi, Pietro Morerio, Andrea Cavallo, Cristina Becchio, and Vittorio Murino. Predicting intentions from motion: The subject-adversarial adaptation approach. *International Journal of Computer Vision*, 128(1): 220–239, 2020.