Journal of Computers Vol. 29 No. 4, 2018, pp. 82-95
doi:10.3966/199115992018082904007

Autoencoder-based Feature Learning from a 2D Depth Map and 3D Skeleton for Action Recognition

Zhi-Ze Wu[1,2*], Shou-Hong Wan[2], Li Yan[2], Li-Hua Yue[2]

[1] Department of Computer Science and Technology, Hefei University, Hefei, 230601, P.R. China
wuzhize@mail.ustc.edu.cn

[2] School of computer Science and Technology, University of Science and Technology of China, Hefei, 233027, P.R. China

Received 15 February 2017; Revised 5 June 2017; Accepted 3 July 2017

Abstract. 3D skeleton is compact for human action recognition. Existing approaches focus mainly on performing action recognition with only joint coordinates. However, they are difficult recognizing some similar or complex actions based on skeleton data only. Furthermore, they could have poor performance when the estimated skeletal joints are not reliable or when the actions share a large overlap in the sequences. Noticing that the existing 3D skeleton data is usually obtained from depth map and a neural network can simulate arbitrary relationships, we propose an efficient approach for action recognition by combining 2D depth information with 3D skeleton information using a new deep architecture called Deep Multimodal Auto-Encoder (DMAE) in this paper. First, the framework of DMAE employs two Auto-Encoders to extract hidden representations for the 2D depth maps and the 3D skeletons. Second, DMAE uses a two-layer neural network to map the hidden representations of the 2D depth maps to those of the 3D skeletons. Third, based on a Back-propagation Neural Network (BP-NN), it can jointly explore the hidden 2D/3D representations and the appropriate relationships between them. We finally use Temporal Pyramid Matching (TPM) on the learned features to generate temporal representations and perform classifications using a linear SVM. Additionally, we demonstrate the effectiveness and efficiency of DMAE by extensive experimentation using two popular action datasets.

Keywords: 2D depth map, 3D skeleton, action recognition, Auto-Encoder, back-propagation neural network

1 Introduction

The preparation of manuscripts which are to be reproduced by photo-offset requires special care. Papers submitted in a technically unsuitable form will be returned for retyping, or canceled if the volume cannot otherwise be finished on time.

Action recognition aims to recognise human actions from videos in a given scenario automatically. It has been an active research field for decades and is one of the fundamental technologies for many applications such as surveillance, video games, and robotics [1-2]. Traditional studies about action recognition focus mainly on recognising actions from monocular RGB videos that are recorded by 2D cameras [3]. Unfortunately, the monocular RGB data are highly sensitive to various factors such as illumination changes, variations in view-point, occlusions, and background clutter [4-5]. Moreover, monocular video sensors can not fully capture human motion in 3D space. However, human actions are usually represented and recognized in 3D space. Therefore, despite significant research efforts over the past few decades, inferring high-level knowledge from a color video, especially in a complex and unconstrained scene, remains a challenging problem.

* Corresponding Author

The human body is regarded as an articulated system of rigid segments (body parts) that are connected by joints, and a human action is composed of the motions of these segments, which are represented by the movements of the human skeleton joints in 3D space. In 1975, Johansson's experiments showed that humans can reognise activities by seeing only light spots that were attached to the person's major joints [6]. Thus, if human skeletons can be reliably extracted, then we can improve the effectiveness of action recognition by classifying the temporal evolution of human skeletons [7]. With the development of depth sensors such as the Kinect [8], some studies that have focused on extracting the 3D skeleton joints from the depth maps produced by depth sensors have been performed [8-9]. As a result, renewed interest in skeleton-based human action recognition is taking place.

Human skeleton-based action recognition usually focuses on extracting the characteristics of body postures and their dynamics over time to represent a human action. We can broadly categorise the existing skeleton-based action recognition approaches into three main groups:

Joint-based representations [10-13]. These approaches try to model motion of either individual joints or combinations of joints using various features like joint position [10-11], joint orientations with respect to a fixed coordinate axis [12], pairwise relative joint positions [13], etc.

Body part-based representations [4, 14-16]. These approaches consider the human skeleton as a connected set of rigid segments and attempt to represent a skeleton by means of the geometric relations among the different body parts. In [4], Vemulapalli represented human skeletons as points in the Lie group and explicitly modeled the 3D geometric relationships between various body parts using rotations and translations. Human body was divided into five different parts in [14], and human actions were represented using the motion parameters of individual body parts. Human skeleton was represented using 3D joint angles in [15], and the temporal evolutions of these angles were compared using DTW. In [16], skeletal sequences were represented using pairwise affinities between joint angles trajectories and then classified using a linear SVM.

Dynamics-based representations [17-19]. These methods focus mainly on model-ling the dynamics of either subsets or all of the joints in the skeleton. This approach can be accomplished by considering linear dynamical systems (LDS) [15, 20] or hidden Markov models (HMM) [19] or recurrent neural networks (RNNs) [20] or mixed approaches [21].

The aforementioned approaches focus mainly on performing action recognition with only joint coordinates. However, they could have poor performance when the estimated skeletal joints are not reliable or when the actions share a large overlap in the sequences. In this paper, we propose an efficient approach for learning skeletal features by jointly combining the 2D depth and 3D skeleton information using a new deep architecture called Deep Multimodal Auto-Encoder (DMAE).

We consider next the studies on unsupervised feature learning [22-23, 34-35], which is a set of algorithms that attempt to learn a hierarchy of features by building high-level features from low-level ones. Some of the models, such as Convolutional Neural Networks (CNN) [25], Deep Belief Nets (DBN) [26] and Auto-Encoders [27] have been shown to yield excellent results on several tasks, e.g., object recognition, natural language processing, and audio classification. One reason for the success of deep learning methods is that they usually learn to capture the posterior distribution of the underlying explanatory factors for the data [28]. Inspired by the learning capability and capacity of the deep learning model, we hypothesised that deep architectures would be perfectly suited to seeking the proper representations for 2D depth maps and 3D skeletons and modelling their relationships.

Taking full advantage of the Convolution Auto-Encoder (CAE) and the Denoising Auto-Encoder (DAE) in learning 2D spatial structure features and resisting 3D noise data, we employ them as the basic units in the DMAE. For the DAE unit, we propose to add an action category and temporal constraints into the model that integrates the temporal information and to add intra-and inter-class information into the learned features, which are more discriminative and able to capture the small but significant differences between actions. By using a two-layer neural network, the DMAE can map the hidden representation of 2D depth maps to those of 3D skeletons. Additionally, it can jointly explore the hidden 2D/3D representations and the appropriate relationship between them based on the Back-propagation Neural Network (BP-NN). In this way, rather than elaborately design the joint-based or part-based local features, we can learn discriminative features that are robust to noisy skeletal data for recognising human actions.

Irrespective of the skeleton representation that is used, we apply Temporal Pyramid Matching (TPM) [6] for temporal modelling. In addition, we use a linear SVM [36] to classify each sequence into an

action category. Experiments show that our proposal achieves superior results using two benchmark datasets.

Contributions. (1) We propose a novel skeleton representation learning framework called DMAE with a 2D depth map. The DMAE explicitly models the relationships between the 3D skeleton and the 2D depth map using a BP-NN. (2) To capture the subtle but significant differences between actions, we integrate the action category and temporal information into the learning architecture. (3) We experimentally show that the proposed approach outperforms various state-of-the-art skeleton-based human action recognition approaches by evaluating it using two different datasets: MSR-Action3D [30] and Florence3D-Action dataset [33]. In addition, our experiments show that our proposal can reconstruct and denoise corrupted data.

Organisation. We describe the framework of deep multimodal Auto-Encoder and the training process in Section 2, and we present the temporal modelling and classification approach in Section 3. Experimental results and discussions are presented in Section 4. Finally, we conclude the paper in Section 5.

2 Deep Multimodal Auto-Encoder

Auto-Encoders (AE) are popular feature learning models that are conceptually simple, easy to train and allow for efficient inference and training. Therefore, we make use of them to explore the representations from 2D depth maps and using them to jointly extract the hidden representations from 3D skeletal data. It can efficiently describe depth maps and skeletons through hidden layers. Hidden layers of maps and skeletons are successively connected by the BP-NN. In this way, a multi-layered deep neural network is constructed. Given a set of samples $X=[x_1,x_2,\cdots,x_n]\in\mathbb{R}^d$, the basic AE aims to minimise the reconstruction error of all of the samples:

$$L_{AE}(\theta)=\sum_i L(x_i,\mathrm{r}), \tag{2.1}$$

where x_i and r are the initial input and the reconstruction output, respectively, and the loss function L is usually a square loss function. The hidden layer implies an encoding process and a decoding process:

$$\begin{cases} f_\theta(x_i)=s(Wx_i+b_1) \\ g_\theta(h_i)=s(W'h_i+b_2). \end{cases} \tag{2.2}$$

Encoder. The deterministic mapping f_θ that transforms an input d-dimensional vector. x. into a hidden layer d'-dimensional feature vector. h. Its typical form is an affine mapping followed by a non-linear function.

Decoder. The resulting hidden representation h is then mapped back to a reconstructed d-dimensional vector r in an input space, $r=g_\theta(h)$. This mapping g_θ is called the decoder. Its typical form is again an affine mapping that is optionally followed by a squashing non-linearity.

The parameter set of this model is $\theta=\{W,W',b_1,b_2\}$, where. W. and W'. are the encoder and decoder weight matrices, b_1 and b_2 are the offset vectors of d' and d dimensionality, respectively. Additionally, $s(.)$ is the activation function, which is a sigmoid function in this paper.

It is worthwhile to mention that the input vector. x_i. and the reconstruction vector r have the same dimension d, and the hidden layer h^i has the dimension d'; thus, the size of W is the same as the size of the transpose of W', which is $d'*d$.

Merely using the model mentioned above, we can handle only a single sample and cannot model the relationships between a pair of samples. In skeleton-based action recognition, we are interested in combining the 2D depth and 3D skeleton information and jointly discovering suitable 2D/3D representations and encoding their relationship. We argue that a better representation should depend on not only the input skeleton but also the internal relationship between the 2D/3D pairs. With this consideration in mind, we have developed the DMAE.

2.1 Architecture of the DMAE

The Deep Multimodal Auto-Encoder (DMAE) has a three-stage architecture, as shown in Fig. 1. The first and third stages employ two auto-encoders, which are called the 2D Auto-Encoder and the 3D Auto-Encoder, for learning the hidden representations of the 2D depths and the 3D skeletons, respectively. The second stage is the representation mapping, which incorporates a two-layer neural network to transform the 2D representation into the 3D representation.

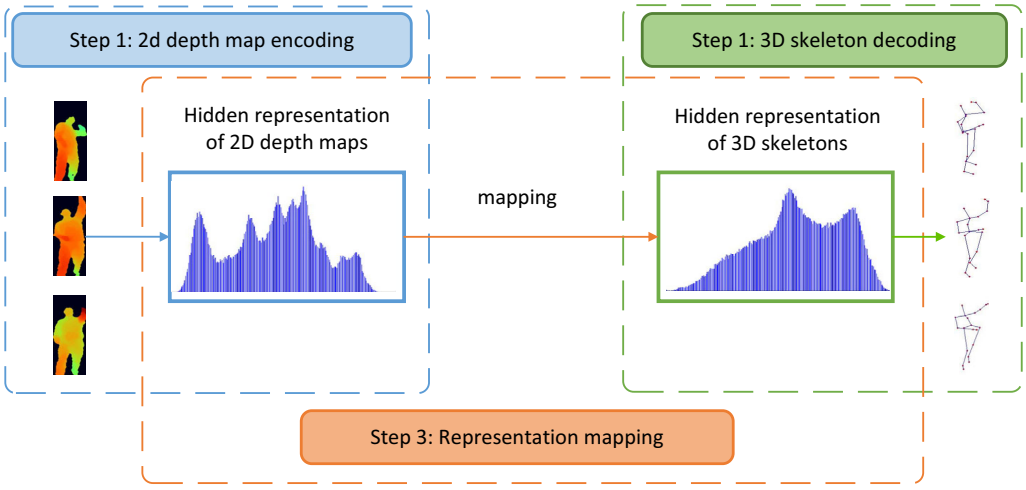


Fig. 1. Flowchart of the DMAE

2D Auto-Encoder. Fully connected AEs and Denoising Auto-Encoders (DAEs) both ignore the 2D image structure. This limitation not only is a problem when addressing realistically sized inputs but also introduces redundancy in the parameters, forcing each feature to be global (i.e., to span the entire visual field). To take account for the 2D spatial structure of the depth maps, we use the Convolution Auto-Encoder (CAE) [29], which also scales well to high-dimensional inputs, to learn non-trivial 2D representations.

The CAE architecture is intuitively similar to the aforementioned AE, except that the weights are shared. For a mono-channel input x, the latent representation of the k-*th* feature map is

$$h_k^{2D} = s\left(x^{2D} * W_k^{2D} + b_k^{2D}\right), \tag{2.3}$$

where the bias is broadcasted to the whole map, and $*$ denotes the 2D convolution. A single bias per latent map is used, because we want each filter to specialise on the features of the whole input (one bias per pixel would introduce too many degrees of freedom). The reconstruction is obtained using:

$$r^{2D} = s\left(\sum_{k\in H} h_k^{2D} * \tilde{W}_k^{2D} + b^{2D}\right), \tag{2.4}$$

where again there is one bias b per input channel. H is the group of latent feature maps; $\tilde{W}$ identifies the flip operation over both dimensions of weights. The cost function to minimise is the mean squared error (MSE), which is similar to equation (2.1).

To create sparsity over the hidden representation, a max-pooling layer is introduced in the CAE. Thus, by erasing all of the non-maximal values in the non overlapping sub-regions, we can force the feature detectors to become more broadly applicable, avoiding trivial solutions such as having only one weight "on" (identity function). Specifically, the encoder of the CAE consists of a convolution layer and a max pooling layer, as shown in Fig. 2. During the reconstruction phase, such a sparse latent code decreases the average number of filters that contribute to the decoding of each pixel, forcing the filters to be more general. Consequently, with a max-pooling layer there is no obvious need for L1 and/or L2 regularisation over the hidden units and/or weights.

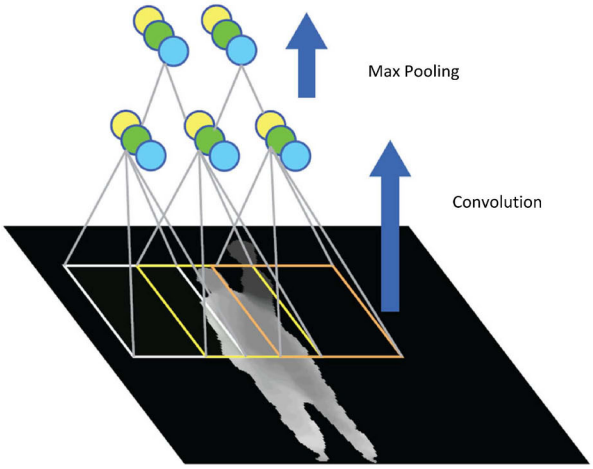


Fig. 2. Convolution and max pooling layer of the CAE

3D Auto-Encoder. We use the Denoising Auto-Encoder (DAE) [24] to learn the 3D skeleton representations. Based on the DAE, we propose to add category information and temporal constraints to make the model capable of emphasising the imparities in different actions, and we term this modified model the Denoising Auto-Encoder with Constraints of Category and Temporal (DAE_CCT).

Fig. 3 shows the architecture of the DAE_CCT. The input x is first corrupted into $\tilde{x}$ using the stochastic mapping $\tilde{x} \sim q(\tilde{x}|x)$ This approach is similar to randomly selecting some of the nodes of the input and blinding them; in other words, every node in the input layer has a possibility q to be switched to zero. The stochastic corrupted data are regarded as the input of the next layer; see Fig. 3. An extra target ct is added to the network, where ct is a vector with length $|l+1|$ Here, we assume l action categories. The first element of vector ct is the current frame's relative temporal location in an action sequence, which is the proportion of the current frame number and the length of the corresponding sequence. The remainder of ct has only one positive element, whose index indicates the action category of the video where the current frame belongs. As a consequence, a constraint vector r_{ct} must be reconstructed by the hidden layer h using a new mapping function $g_{\theta c}$. Similarly, r_x is the reconstruction vector of x by the mapping function $g_{\theta x}$. The new training objective of DAE_CCT is

$$L_{DAE\_CCT}(\theta) = \sum_i \mathrm{E}_{q(\tilde{x}|x)}\left[L\left(x^{(i)}, g_{\theta x}\left(f_\theta\left(x^{(i)}\right)\right)\right) + \lambda L\left(c^{(i)}, g_{\theta c}\left(f_\theta\left(x^{(i)}\right)\right)\right)\right],. \quad \textbf{(2.5)}$$

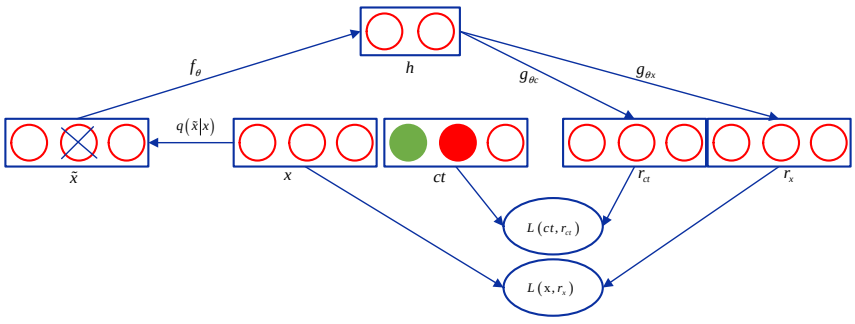


Fig. 3. Architecture of the DAE_CCT. The green solid circle in ct indicates the temporal information, and the remainder is a standard unit vector, which indicates the category of the video where the current frame belongs

where $\mathrm{E}_{q(\tilde{x}|x)}[\cdot]$ is the expectation over the corrupted examples $\tilde{x}$ drawn from the corruption process $q(\tilde{x}|x)$, and λ is a hyper-parameter that controls the strength of the action category and the temporal regularisation. It can be optimised by stochastic gradient descent, similar to the process of optimising the traditional AEs.

Note that we use a regularisation term rather than directly learn the action category labels as targets. This choice is made because the input is the skeleton joint vector for each frame, which means that the category labels are for the whole sequence. Apparently, there are some similar postures among the actions. For example, the *"stand"* and *"put the hands down"* human-postures appear at the beginning of most actions. Training the same human-posture for different category labels will lead to trivial results. The regularisation term establishes a trade-off between reconstructing the input data and preserving the action category information as well as the temporal information.

Representation mapping. By stacking several layers of 2D Auto-Encoders and 3D Auto-Encoders, we can build deep Auto-Encoders that have great expressive power and finally generate the hidden representations h_i^{2D} and h_i^{3D}. After obtaining the 2D/3D hidden representations, the Neural Network realises mappings from the 2D hidden representation to the 3D hidden representation. We denote the parameters in this stage as (W^N, b^N), ,where W^N is the weight matrix, and b^N is the bias term. The mapping function becomes

$$h_i^{3D} = s\left(W^N h_i^{2d} + b^N\right). \tag{2.6}$$

The construction of the DMAE indicates that the model is simple and flexible. Auto-Encoders ensure that the hidden representations describe 2D/3D data well and the neural network can learn complex relationships between the 2D/3D representations; notably, the mapping function and the hidden representations are jointly optimised and are thus correlated.

2.2 Training Procedure of the DMAE

The DMAE must explore the 2D/3D hidden representations and simultaneously connect them using a well-trained mapping function based on a BP-NN. BP-NN is trained with automatic optimisation. For this purpose, we design a two-part training procedure: the first part is the initialisation (Stages 1, 2, and 3 in Fig. 4), and the second part is the fine-tuning, which is implemented in Stage 4.

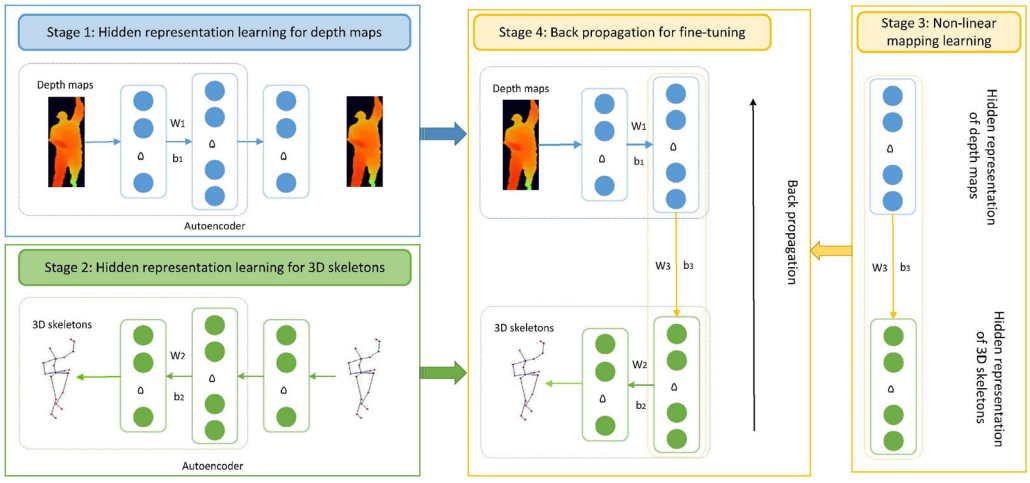


Fig. 4. The process of learning a non-linear mapping

Initialisation. To train the DMAE, the hidden representations of the 2D/3D inputs are first generated. According to the 2D Auto-Encoder introduced in Section 2.1, given the 2D depth maps, and based on Equations (2.4) and (2.5), the 2D hidden representation can be obtained by minimizing the following:

$$L^{2D} = \sum_{i}^{n} \left\| X_i^{2D} - R_i^{2D} \right\|^2, \tag{2.7}$$

which is the reconstruction error of the 2D Auto-Encoder. Similarly, for the 3D counter-part, the corresponding hidden representation is induced by minimising:

$$L^{3D} = L_{DAE\_CCT}. \tag{2.8}$$

In the training of the DMAE, neural networks (NNs) are applied to learn the mapping. Based on the obtained hidden representations h_i^{2D} and h_i^{3D}, the optimisa-tion of the neural network in Stage 3 learns the mapping function by minimising:

$$\sum_{i}^{n} \left\| s\left(W^N h_i^{2D} + b^N\right) - h_i^{3D} \right\|^2. \tag{2.9}$$

To minimise the average squared error in (2.9), back propagation (BP) is introduced to adjust the weights and thresholds of the network. By integrating (2.7), (2.8), and (2.9), we can obtain the whole objective function:

$$L_{2D} + L_{3D} + \sum_{i}^{n} \left\| s\left(W^N h_i^{2D} + b^N\right) - h_i^{3D} \right\|^2. \tag{2.10}$$

It should be stressed that during the initialisation, the parameters $\left(W^{2D}, b^{2D}\right)$ and $\left(W^{3D}, b^{3D}\right)$ are used only to encode and decode the input 2D and 3D features, while $\left(W^N, b^N\right)$ only encodes the mapping from h^{2D} to h^{3D}. To achieve a complete training procedure for the DMAE, we discuss how to connect these parameters.

Parameter tuning. The above initialisation process does not guarantee that the parameters obtained are optimal for skeleton based action recognition, because individually training each set of parameters could lead to a high reconstruction error when they are combined. Therefore, all of the parameters must be refined via a connected optimizstion.

Given the initialised parameters $\left(W_1^{2D}, W_1^{3D}, W^N, b_1^{2D}, b_1^{3D}, b^N\right)$, because the output of the DMAE is R_i, the fine-tuning objective function is similar to that in (2.11) and (2.1):

$$L_{DMAE} = \sum_{i}^{n} \left\| X_i - R_i \right\|^2. \tag{2.11}$$

The involved values R_i, h_i^{2D} and h_i^{3D} follow (2.6), (2.7), and (2.8) respectively. As with most deep learning methods, we adopt BP to optimise (2.11). The DMAE training steps are summarised in Algorithm 1. Back-propagation is applied to deep neural network sequentially. The order of optimisation is 2D maps to a 2D hidden representations, the mapping from 3D skeletons to 3D hidden representations and the mapping from 2D hidden representations to 3D hidden representations. The error of one layer is back propagated only to its previous layer.

Algorithm 1. *Details of the DMAE Training Procedure With BP-NN*

Input: 2D depth map set X ; 3D skeleton set Y .

Output: Hidden representation H^{3D} .

Stage 1

Compute hidden representation of X with the 2D Auto-Encoder defined in (2.8);

Stage 2

Compute hidden representation H^{3D} of Y with the 3D Auto-Encoder defined in (2.9);

Stage 3

Compute BP-NN mapping A with H^{2D} and H^{3D};

Stage 4

---

repeat

Compute backward result R^{3D} *with* A *and* H^{3D}*;*

Compute L_{DMAE} *in (2.11);*

Update the network weights $\left(W^N, b^N\right)$*;*

until L_{DMAE} *in (2.11) converges*

3 Temporal Modelling and Classification

As the focus of this paper is the skeleton representation learning, we explicitly model temporal dynamics of the extracted representations with a temporal pyramid matching (TPM) [6]. Motivated by Spatial Pyramid Matching (SPM) [21], the TPM uses a max pooling function to generate the multiscale structure. We recursively partition the video sequence into increasingly finer segments along the temporal direction and use the max pooling to generate histograms from each sub-region. Typically, 3 levels with each containing 1, 2, and 4 segments are used. The final feature is the concatenation of histograms from all of the segments. We illustrate this process in Fig. 5.

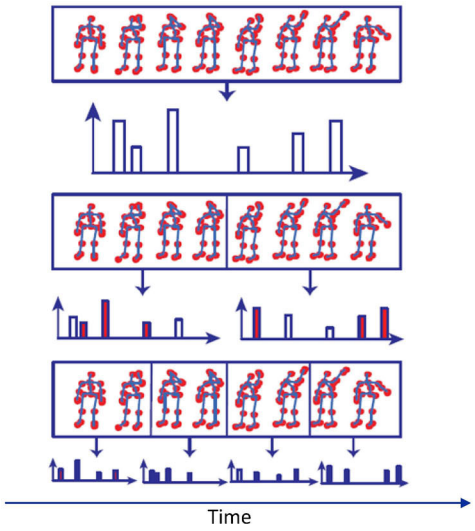


Fig. 5. Temporal pyramid matching

Specifically, we assume that we obtain a feature matrix $U \in \mathbb{R}^{D*T}$ for a video by the DMAE, where T is the frame amount, and D is the dimension for each feature vector. We use z^{ij} to denote the *j*-th feature vector of the *i*-th layer in the pyramid. For the top layer, there is only one vector $z^{11} = \left[z_1^{11}, z_2^{11}, \cdots, z_D^{11}\right]$, which can be computed using

$$z^{11} = F(\mathrm{U}). \tag{3.1}$$

F is the max pooling function, as follows:

$$z_i^{11} = \max\left\{|U_{i1}|, |U_{i2}|, \cdots, |U_{iT}|\right\}, \ \forall i = 1, 2, \cdots D. \tag{3.2}$$

When getting the first layer of the pyramid, we partition U into U_1 and U_2, where $U_1 = \left[u_1, u_2, \cdots u_{T/2}\right]$, and $U_2 = \left[u_{T/2+1}, u_{T/2+2}, \cdots u_T\right]$. Thus, we can obtain the feature vectors for the second layer with the max pooling function. With that analogy, we can obtain the third layer and the *n*-th layer, and finally, obtain the representation $z = \left[z^{11}, z^{21}, z^{22}, \cdots, z^{nm}\right]$ by connecting the vectors from all of the layers, where $m = 2^{n-1}$, for each video.

After obtaining the final representations, we use a multi-class linear SVM to speed up the training and

testing. The experimental results will be discussed in the next section.

4 Experiments

In this section, we evaluate our proposal and compare it with several recent studies on two benchmark datasets, namely, MSR-Action3D [30] and Florence3D-Action [33]. We also reveal the denoising capability of our method for reconstructing noisy 3D joint sequences.

MSR-Action3D. This dataset captured by a Microsoft Kinect-like depth sensor, which is widely used in action recognition. This dataset consists of 20 actions performed by 10 different subjects. Each subject performed every action in an unconstrained way two or three times. Altogether, there are 557 valid action sequences with 22077 frames. All of the sequences are captured in 15 FPS, and each frame in a sequence contains 20 skeleton joints. The low accuracy of the skeleton joint coordinates and the presence of many highly similar actions make this dataset very challenging.

Florence3D-Action. This dataset was collected at the University of Florence using a Kinect camera. It includes 9 actions: "wave", "drink from a bottle", "answer phone", "clap", "tight lace", "sit down", "stand up", "read watch", and "bow". Each action was performed by 10 subjects two or three times, which resulted in a total of 215 action sequences. In each frame of the sequence, the 3D locations of the 15 joints are provided with the dataset. Although, these 9 actions are simple, the presence of actions such as "drink from a bottle" and "answer phone" are quite similar to one another. Hence, this dataset is challenging because of the high intra-class variations.

Basic preprocessing. For the skeletal data, we normalised all of the skeleton joints of the above two evaluation datasets to a unified coordinate system. The origin of the coordinate system is the hip centre. To make the skeletons scale-invariant, we selected one referenced skeleton in the dataset and normalised all other skeletons without changing their joint angles, as Vemulapalli did in [4]. Furthermore, we rotated the both ends of the hip to parallel a global x-axis, which makes skeletons view-invariant. For the depth data, we first calculate the mean value μ and the standard deviation σ for each depth map. We then updated each non-zero value d using the following:

$$d = \max\left(\min\left(d, \mu + 2\sigma\right), \mu - 2\sigma\right). \tag{4.1}$$

By this approach, we can eliminate abnormal depth values, and obtain the maximum $d_{\max}$ and the minimum $d_{\min}$. According to the characteristics of the sigmoid, we further scaled all of the non-zero depth values into 0.3 and 0.9, with the following:

$$d = \frac{0.6 * d + 0.3 * d_{\max} - 0.9 * d_{\min}}{d_{\max} - d_{\min}}. \tag{4.2}$$

Based on this preprocessing, we can make the depth differences between the different body parts be more obvious, and we can remove some of the noise points.

4.1 Evaluation Settings and Parameters

For the MSR-Action3D dataset, we follow the cross-subject test settings of [30]. In [33], the dataset was divided into the subsets AS1, AS2, and AS3, each of which consists of 8 actions, as shown in Table 1, and we performed recognition on each subset separately. For this dataset, the first 2D Auto-Encoder has 9 feature maps with a kernel size of 13*13, and the second has 16 feature maps with a kernel size of 6*6; the scaling factors in both of the max pooling layers are 4. In addition, the first 3D Auto-Encoder contains 200 nodes in the hidden layer, and the second contains 400 nodes.

For Florence3D-Action, we leave out each subject from the training set and repeat an experiment for each of them (leave-one-subject-out) by following the experimental protocol proposed in [33]. The first 2D Auto-Encoder has 12 feature maps with a kernel size of 9*9, and the second has 10 feature maps with a kernel size of 7*7; the scaling factors in both of the max pooling layers are 4. The first 3D Auto-Encoder contains 160 nodes in the hidden layer, the second contains 320 nodes.

Table 1. Action sets of the MSR-Action3D dataset

AS1	AS2	AS3
horizontal wave	high wave	high throw
hammer	hand catch	forward kick
forward punch	f draw X	side kick
high throw	draw tick	jogging
hand clap	draw circle	tennis swing
bend	two hands wave	tennis serve
tennis serve	side boxing	golf swing
pickup & throw	forward kick	Pickup & throw

In all of the experiments, we train a deep architecture stacked by 2D Auto-Encoders, 3D Auto-Encoders, and a Neural Network for each dataset, using BP-NN. We penalise the average output $\overline{h}_i$ of the second 3D Auto-Encoder and push it to 0.1, to add some sparsity to the model and learn an over-completed representation of the joint features. The parameter λ in the DAE_CCT is experimentally assigned to 1.5. A four-level TPM, with each containing 1, 2, 4, 8 segments, is used to model the temporal aspect. The multi-class linear support vector machine algorithm here we use is the more commonly used LIBSVM [36], where the value of the penalty factor *C* is determined by the cross validation method. All the results reported in this paper were averaged over ten different combinations of training and test data. In addition, it should be noted that the results of all of the comparative methods on the two datasets are from their corresponding papers.

4.2 Results

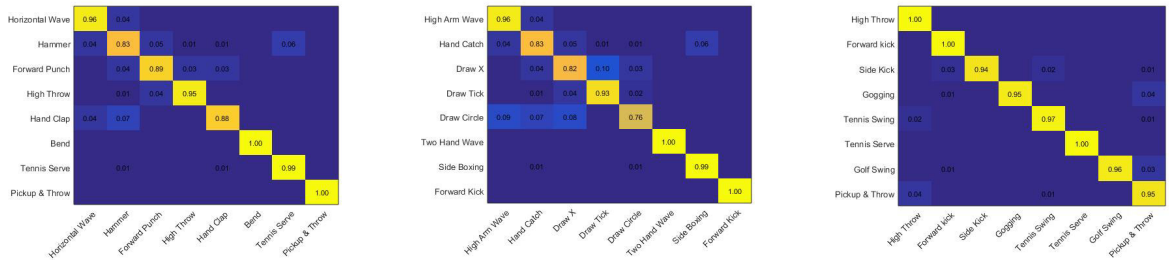
Comparison with the state-of-art. We compare the proposed method DMAE with various state-of-the-art skeleton based human action recognition approaches. Table 2 shows the comparison results on MSR-Action3D and Florence3D-Action. We can see that the DMAE gives excellent results on all of the datasets. It performs comparably to the best result of 94.49% [5] with an accuracy of 94.14% on MSR-Action3D using the protocol [30]. In [5], Du et al. present a hierarchical recurrent neural network, which employs five bidirectional recurrent neural networks as its basic units for skeleton based action recognition. This model is corresponding five human body parts, which has integrated the manual intervention. However, DMAE focuses on the jointly learning based on the hidden representations of 2D depth map and 3D skeleton data. It can get the non-linear mapping between these two kinds of data, also the optimized representations for action recognition or other applications. On Florence3D-Aciton dataset, DMAE is the best with an accuracy of 91.00%.

Table 2. Comparison with state-of-the-art methods

MSR-Action3D dataset (protocol [30])	
Li et al., 2010 [30]	74.7
Mohamed et al.,2013 [12]	90.53
Chen et al.,2013 [8]	90.47
Gowayyed et al.,2013 [31]	91.26
Vemulapalli et al., 2014 [4]	92.46
Du et al., 2015 [5]	94.49
DMAE	94.14
Florence3D-Action dataset (protocol [33]	
Seidenari et al., 2013 [33]	82.00
Devanne et al., 2015 [32]	87.04
DMAE	**91.00**

Fig. 5 shows the confusion matrices for MSR-Action3D AS1, MSR-Action3D AS2, and MSR-Action3D AS3. Fig. 6 shows the confusion matrix for Florence3D-Action. Better performance on subsets AS_1 and AS_2 indicates that the proposed representation is better than the others in differentiating similar actions. Better performance on subset AS_3 indicates that the proposed representation is better than the others in modelling complex actions. We can see that the misclassifications occur mainly among several

very similar actions. For example, in Fig. 5(b), the action “Draw X” is often misclassified to “Draw Tick”, while the action “Draw Circle” is misclassified to “High Arm Wave”. In fact, these two pairs of actions share a large overlap in their sequences.



(a) AS1 - 93.83 (b) AS2 - 91.16 (c) AS3 - 97.43

Fig. 5. Confusion matrices of DMAE on the MSR-Action3D dataset

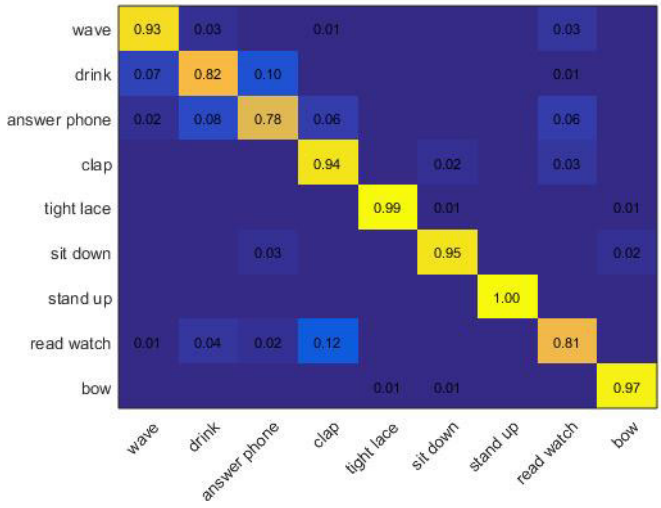


Fig. 6. Confusion matrices of DMAE on Florence3D-Action

Experiments on BP-NN based mapping. Fig. 7 shows an illustration of a comparison between DMAE and DAE_CCT for each action in the Florence3D-Action dataset using the protocol of [33]. The superiority of DMAE is apparent for most of the actions, especially the very similar ones, such as “drink” and “answer phone”. The average recognition accuracy rate is significantly improved by 5.26% from 85.74% after jointly combining the 2D depth information using the BP-NN mapping. It verifies our proposals that skeleton is not enough to recognise some similar or complex actions and the combination with depth map using a neural network is an efficient way to mitigate such problem.

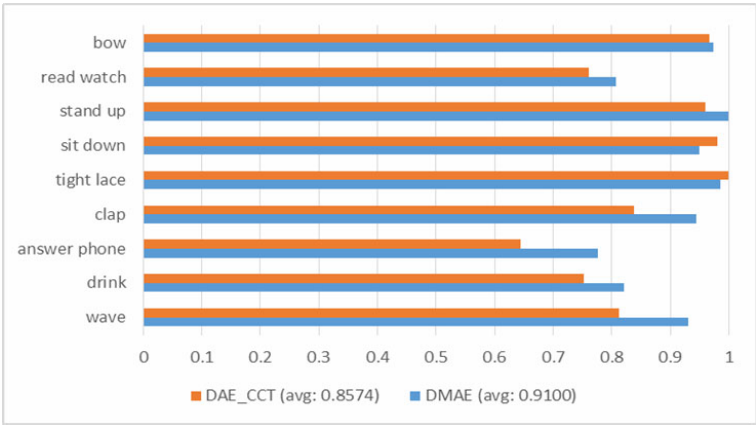


Fig. 7. Comparison of DMAE and DAE_CCT for each action in the Florence3D-Action dataset

Capability to denoise the corrupted data. The proposed model also has the strong capability of reconstructing realistic data from corrupted input. The top row of Fig. 8 is an action sequence "high arm wave", which was selected from the MSR-Action3D dataset. To better demonstrate our algorithm efficiency, we added some Gaussian noise to the joint positions and left out the joints randomly. The bottom row is the reconstructed action sequence, where we can observe that the missing joints are well restored via our model and the motions are more natural and fluent than before.

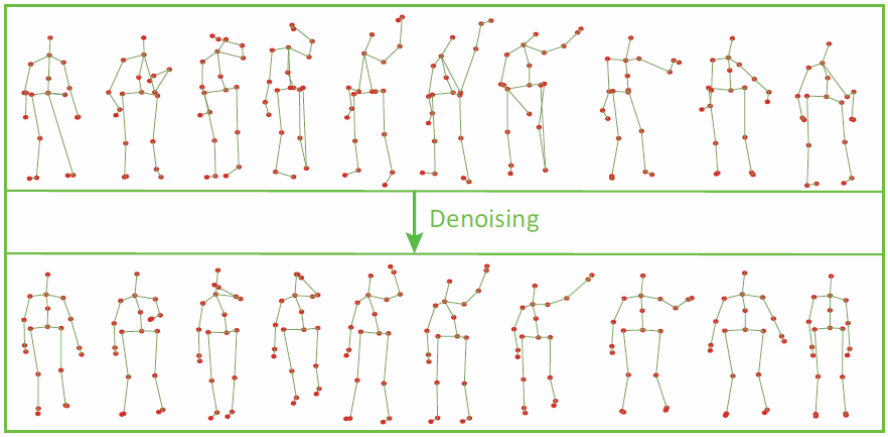


Fig. 8. Examples that shows the capability of DMAE to denoise the corrupted data

5 Conclusions

In this paper, a new framework for skeleton-based action recognition is proposed. This framework jointly combines the 2D depth and 3D skeleton information using a new deep architecture called Deep Multimodal Auto-Encoder (DMAE). Taking full advantage of the Convolution Auto-Encoder (CAE) and the Denoising Auto-Encoder (DAE) in learning 2D spatial structure features and resisting 3D noise data, we employ them as the basic units in the DMAE. Moreover, the DMAE uses a two-layer neural network to map the hidden representation of 2D depth maps to those of 3D skeletons. Additionally, it can jointly explore the hidden 2D/3D representations and the appropriate relationships between them, which is helpful in making up for deficiencies in the noisy skeletal data. With the skeleton representations being learned, we use the Temporal Pyramid Matching (TPM) to model the temporal information and employ the linear SVM for classification. Experiments using the MSR-Action3D dataset and the Florence3D-Action show that our proposal achieves superior results in terms of its action recognition performance.

In future work, we will extend this framework to explore the nonlinear relationship between other types of data, such as RGB, and further study the action recognition based on an effective end-to-end solution for modelling the temporal dynamics from the perspective of the whole action sequences.

References

[1] J. Wang, Z. Liu, Y. Wu, J. Yuan, Mining actionlet ensemble for action recognition with depth cameras, in: Proc. the 25th International Conference of Computer Vision and Pattern Recognition (CVPR), 2012.

[2] X. Yang, C. Zhang, Y. Tian, Recognizing actions using depth motion maps-based histograms of oriented gradients, in: Proc. the 20th ACM international conference on Multimedia, 2012.

[3] K.J. Aggarwal, S.R. Michael, Human activity analysis: A review, ACM Computing Surveys 43(3)(2011) 16.

[4] R. Vemulapalli, A. Felipe, C. Rama, Human action recognition by representing 3d skeletons as points in a lie group, in: Proc. the 27th International Conference on Computer Vision and Pattern Recognition,2014.

[5] Y. Du, W. Wang, L. Wang, Hierarchical recurrent neural network for skeleton based action recognition, in: Proc. the 26th International Conference on Computer Vision and Pattern Recognition, 2015.

[6] G. Johansson, Visual motion perception. Scientific American 232(6)(1975) 76-88.

[7] M. K. Knutzen, Kinematics of human motion, Wiley Online Library (1998) 808-809.

[8] S. Toby, K. Alex, F. Andrew, F. Mark, B. Andrew, C. Mat, M. Richard, Real-time human pose recognition in parts from single depth images, Communications of the ACM 56(1)(2013) 116-124.

[9] J. Jonathan, J. Arjun, L. Yann, B. Christoph, Joint training of a convolutional network and a graphical model for human pose estimation, in: Proc. the 27th International Conference on Advances in Neural Information Processing Systems, 2014.

[10] X.-D.Yang, Y.-L. Tian, Eigenjoints-based action recognition using naive-bayes-nearest-neighbor, in: Proc. 25th International Conference on Computer Vision and Pattern Recognition, 2012.

[11] J. Wang, Z.-C. Liu, Y. Wu, J.-S. Yuan, Mining actionlet ensemble for action recognition with depth cameras, in: Proc. 25th International Conference Computer on Vision and Pattern Recognition, 2012.

[12] E.M. Hussein, T. Marwan, A.-G. Mohammad, E.-S. Motaz, Human action recognition using a temporal hierarchy of covariance descriptors on 3D joint locations, in: Proc. 23th International Joint Conference On Artificial Intelligence, 2013.

[13] A. Eweiwi, S.-C. Muhammed, B. Christian, G. Juergen, Efficient pose-based action recognition, in: Proc. the 12th Asian Conference on Computer Vision, 2014.

[14] C.-Y Wang, Y.-Z. Wang, L.-Y. Alan, An approach to pose-based action recognition, Proc. the 26th International Conference on Computer Vision and Pattern Recognition, 2013.

[15] R. Chaudhry, O. Ferda, K. Gregorij, B. Ruzena, Bio-inspired dynamic 3D discriminative skeletal features for human action recognition, in: Proc. the 26th International Conference on Computer Vision and Pattern Recognition, 2013.

[16] E. Ohn-Bar, M.-M. Trivedi, Joint angles similarities and HOG2 for action recognition, in: Proc. the 26th International Conference on Computer Vision and Pattern Recognition, 2013.

[17] Y. Du, W. Wang, L. Wang, Hierarchical recurrent neural network for skeleton based action recognition, in: Proc. the 26th International Conference on Computer Vision and Pattern Recognition, 2015.

[18] J.-J. Luo, W. Wang, H.-R. Qi, Group sparsity and geometry constrained dictionary learning for action recognition from depth maps, in: Proc. 2013 IEEE International Conference on Computer Vision, 2013.

[19] D. Wu, S. Ling, Leveraging hierarchical parametric networks for skeletal joints based action segmentation and recognition, in: Proc. the 27th International Conference on Computer Vision and Pattern Recognition, 2014.

[20] R. Slama, W. Hazem, D. Mohamed, Accurate 3D action recognition using learning on the Grassmann manifold, Pattern Recognition 48(2)(2015) 556-567.

[21] L.-L. Presti, L.-C. Marco, S. Stan, C. Octavia, Gesture modeling by hanklet-based hidden markov model, in: Proc. the 12th Asian Conference on Computer Vision, 2014.

[22] Y. Bengio, Learning deep architectures for AI, Foundations and trends® in Machine Learning 2(1)(2009) 1-127.

[23] J. Martens, S. Ilya, Learning recurrent neural networks with hessian-free optimization, in: Proc. the 28th International Conference on Machine Learning, 2011.

[24] P. Vincent, L. Hugo, L. Isabelle, B. Yoshua, P.-A. Manzagol, Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion, Journal of Machine Learning Research 11(2010) 3371-3408.

[25] L.-C., Yann, B. Leon, B. Yoshua, H. Patrick, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86(11)(1998) 2278-2324.

[26] G.E. Hinton, O. Simon, Y.-W. The, A fast learning algorithm for deep belief nets, Neural computation 18(7)(2006) 1527-

1554.

[27] G.E. Hinton, R.R. Salakhutdinov. Reducing the dimensionality of data with neural networks, Science 313(5786)(2006) 504-507.

[28] Y. Bengio, C. Aaron, V. Pascal, Representation learning: a review and new perspectives, IEEE Transactions on Pattern Analysis and Machine Intelligence 35(8)(2013) 1798-1828.

[29] J. Masci, M. Ueli, C. Dan, S. Jürgen, Stacked convolutional auto-encoders for hierarchical feature extraction, in: Proc. the 20th International Conference on Artificial Neural Networks, 2011.

[30] W.-Q. Li, Z.-Y. Zhang, Z.-C. Liu, Action recognition based on a bag of 3d points, in: Proc. the 23th International Conference on Computer Vision and Pattern Recognition, 2010.

[31] M.A. Gowayyed, T. Marwan, E.H. Mohammed, E.-S. Motaz, Histogram of oriented displacements (HOD): describing trajectories of human joints for action recognition, in: Proc. 23th International Joint Conference On Artificial Intelligence, 2013.

[32] M. Devanne, W. Hazem, B. Stefano, P. Pietro, D. Mohamed, D.B. Alberto, 3-D human action recognition by shape analysis of motion trajectories on Riemannian manifold, IEEE Transactions on Cybernetics 45(7)(2015) 1340-1352.

[33] L. Seidenari, V. Vincenzo, B. Stefano, B. Alberto, P. Pietro, Recognizing actions from depth cameras as weakly aligned multi-part bag-of-poses, in: Proc. the 26th International Conference on Computer Vision and Pattern Recognition, 2013.

[34] A. Budiman, M.I. Fanany, C. Basaruddin, Stacked denoising autoencoder for feature representation learning in pose-based action recognition, in: Proc. the 3rd Global Conference on Consumer Electronics (GCCE), 2014.

[35] Y. Gao, R.-G. Ji, X.-B. Gao, P.-M. Jodoin. Signal processing and learning methods for 3D semantic analysis. Signal Processing 112(C)(2015) 1-3.

[36] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines, ACM Transactions on Intelligent Systems and Technology 2(3)(2011) 27.