Hindawi
Advances in Multimedia
Volume 2018, Article ID 5141402, 10 pages
https://doi.org/10.1155/2018/5141402

Research Article

Indian Classical Dance Action Identification and Classification with Convolutional Neural Networks

P. V. V. Kishore, K. V. V. Kumar, E. Kiran Kumar, A. S. C. S. Sastry, M. Teja Kiran, D. Anil Kumar, and M. V. D. Prasad

Department of Electronics and Communication Engineering, KL University, Vaddeswaram, Guntur, India

Correspondence should be addressed to P. V. V. Kishore; pvvkishore@kluniversity.in

Received 13 October 2017; Accepted 20 December 2017; Published 22 January 2018

Academic Editor: Lin Wu

Extracting and recognizing complex human movements from unconstrained online/offline video sequence is a challenging task in computer vision. This paper proposes the classification of Indian classical dance actions using a powerful artificial intelligence tool: convolutional neural networks (CNN). In this work, human action recognition on Indian classical dance videos is performed on recordings from both offline (controlled recording) and online (live performances, YouTube) data. The offline data is created with ten different subjects performing 200 familiar dance mudras/poses from different Indian classical dance forms under various background environments. The online dance data is collected from YouTube for ten different subjects. Each dance pose is occupied for 60 frames or images in a video in both the cases. CNN training is performed with 8 different sample sizes, each consisting of multiple sets of subjects. The remaining 2 samples are used for testing the trained CNN. Different CNN architectures were designed and tested with our data to obtain a better accuracy in recognition. We achieved a 93.33% recognition rate compared to other classifier models reported on the same dataset.

1. Introduction

Automatic human action recognition is a complicated problem for computer vision scientists, which involves mining and categorizing spatial patterns of human poses in videos. Human action is defined as a temporal variation of human body. The last decade has seen a jump in online video creation and the need for algorithms that can search within the video sequence for a specific human pose or object of interest. The problem is to extract and identify a human pose and classify it into labels based on trained CNN feature maps. The objective of this work is to extract the feature maps of Indian classical dance poses from both online and offline data.

However, the constraints are video resolution, frame rate, background lighting, scene change rate, and blurring to name a few. The analysis on online content is a complicated process as the most of the users end up uploading the videos with poor quality, which shows all the constraints as a hindrance in automation of video object segmentation and classification. Dance video sequences online are having far many constraints for smooth extraction of human dance features. Automatic dance motion extraction is complicated due to complex poses and actions performed at different speeds in sink to music or vocal sounds. Figure 1 shows a set of online and offline (lab captured) Indian classical dance video frames for training and testing the proposed CNN algorithm.

The created offline dataset is having 200 Indian classical dance mudras/poses performed by 10 native classical dancers (i.e., 10 sets) at a rate of 30 frames per second (fps). Training is initiated with three different batch sizes. In Batch I of training there is only one set, that is, 200 poses performed by 1 dancer for 2 seconds each at 30 fps, total of $200 \times 1 \times 2 \times 30 = 12000$ dance pose images. Batch II of training is done using 5 sets, that is, a total of $200 \times 5 \times 2 \times 30 = 60000$ dance pose images. In Batch III of training, 8 sets of sign images were used. The trained CNNs are tested with two discrete video sets having different dance performers with varying backgrounds. The robustness testing is performed in two cases. In Case I of testing, the same dataset, that is, an already trained dataset, is used and in Case II of testing different dataset is used. The similar training and testing are done on online data also. Figure 1 shows the sample database created for this work. The

Figure 1: Sample Indian classical dance from online and offline datasets used in this work.

performance of the CNN algorithms is measured for both online and offline data based on their accuracy in recall and recognition rates.

2. Literature Review

Local information of the human in the video is the popular feature for action identification and classification in recent times. This section focuses on giving a current trend in human action recognition and how it is used in recent works for classifying dance performances. The human action recognition is subdivided into video object extraction, feature representation, and pattern classification [1, 2]. Based on these models, numerous visual illustrations have been proposed for discriminating human action based on shape templates in space-time [1], shape matching [2], interest points in 2D space-time models [3], and representations using motion trajectories [4]. Dense trajectory based methods [5] have shown good results for action recognition by tracking sampled points through optical flow fields. Optical flow fields are based on preconditioning on brightness and object motion in a video [6] by assuming uniform brightness variations and object motions in consecutive frames. Finding such a video having uniform brightness happens in a movie or in a lab setup. Hence, these approaches need to be robust in estimating human actions. Data driven methods with multiple feature fusion [7] with artificial intelligence models [8] are currently being explored with the increase in computing power.

Indian classical dance forms are practiced for 5000 years worldwide. However, it is difficult for a dance lover to fully hold the content of the performance as it is made up of hand poses, body poses, leg movements, hands with respect to face and torso, and finally facial expressions. All these movements should synchronize in precision with both vocal song and the corresponding music for various instruments. Apart from these complications, the dancer wears complicated dresses with a nice makeup and at times during performance the backgrounds are changing depending on the story which truly makes this an open-ended problem. Mohanty et al. [9] highlights the difficulties in using state-of-the-art pose estimation algorithms such as skeleton estimation [10] and pose estimation [11] failing to track the dancers moves in both offline and online videos. Samanta et al. [12] used histogram of oriented optical flow (HOOF) features with sparse representations. In our previous work, we approached the problem with SVM classifier [13] on dance videos and found that only multiclass SVMs should be considered. The average recognition rates obtained in implementing Adaboost [14], Artificial Neural Networks (ANN) [15], deep ANN [16], and adaptive graph matching (AGM) [17] on dance data are not up to the mark.

In recent research, application of deep learning in object recognition is most suitable. CNN is powerful in solving most computer vision based tasks [18–22] such as object recognition [23] and classification [24]. Classifying at faster rate on a huge dataset is a complicated problem. Without the knowledge of an expert using deep hidden layers CNN extracts image information and avoids the process of complex feature extraction. The first deep CNN is introduced by LeCun et al. [25] with two convolutional layers. As the image datasets become of large scale, this requires a wider and deeper CNN such as ImageNet [26]. By increasing the depth of the network further Simonyan and Zisserman [27] proposed VGGNet.

Andrew Ng et al. have performed fundamental research on CNNs to achieve improved performance of CNN algorithms and structural optimization [28–31]. LeCun et al. in [25] highlighted that deep CNN is a breakthrough in image, video, audio, and speech processing. So far, no extensive research has been done which explores deep CNN for Indian classical dance classification. The aim of this paper is to bring out the CNN performance in recognizing the mudras/poses of ICD.

Deep learning methods can also be used in evaluating the ICD classification system. However, still there is a great space for further improvements. Deep CNN is suitable for giving solutions to complex problems with a huge quantity of data [24]. For example, the classification accuracy is improved in ImageNet dataset [24] which has 1.2 million images almost covering 1000 categories. In such cases, we need to consider how to take advantage of CNN.

With convolutional neural networks, we need to consider how to design and train a network that adapts to various objects. The major problem to be solved is with the quality and sizes of the images. The unbalanced amounts of low and high quality images in the dataset lead to the unbalanced classification.

The motivation for implementing the deep CNN model for Indian classical dance identification is that the feature learning in CNNs is highly automated from the input images, avoiding the complexity in extracting the various features for traditional classifiers for sign recognition. Through the deep architecture, the learned features are deemed as the higher level abstract representation of low level sign images. Hence, we develop the deep CNN model for Indian classical dance identification in this paper.

In this paper, a novel CNN based Indian classical dance identification method is proposed to achieve higher recognition rates. Different CNN architectures are implemented and tested on our dance data to bring out the best architecture for recognition. Three different pooling techniques, namely, mean pooling, max pooling, and stochastic pooling, are implemented and stochastic pooling was found to be the best for our case. To prove the capability of CNN in recognition, the results are compared with the other traditional state-of-the-art techniques SVM, AGM, Adaboost, ANN, and deep ANN.

The rest of the paper is as follows: in Section 3, the proposed architecture of CNN is described. Section 4 discusses the results obtained in different cases. Finally, Section 5 concludes the outcomes of this paper.

3. System Architecture

We designed our multistage CNN model by acquiring knowledge following [21, 25]. The model is constructed with input layer, four convolutional layers, five rectified linear units (ReLu), and two stochastic pooling layers, one dense and one SoftMax output layer. Figure 2 shows the proposed system architecture.

The proposed CNN architecture uses four convolutional layers with different window sizes followed by an activation function and a rectified linear unit for nonlinearities. The convolutional windows are of sizes 16×16, 9×9, 5×5, and 5×5 from layers 1 to 4, respectively. Three kinds of pooling strategies were tested via mean pooling, max pooling, and stochastic pooling and stochastic pooling was found to be suitable for our application. The feature representation is done by considering two layers of stochastic pooling. Only two layers of pooling are initiated to avoid a substantial information loss in feature representation. Classification stage is implemented with dense/fully connected layers followed by activation functions. SoftMax regression is adopted in classification.

Indian classical dance video frames of size 640×480 are taken as an input to the system. As a first step, the frames are preprocessed by resizing them to $128 \times 128 \times 3$. Resizing of input video frames will increase the computational capability of the high-performance computing (HPC) on which the program is being implemented. The HPC used for training the CNN is a 6-node combined CPU-GPU processing machine.

Table 1: Layer information and parameters of CNN.

Layer (type)	Function	Output shape
Input		$3 \times 640 \times 480$
conv1	Convolution	$16 \times 128 \times 128$
activation_1	Activation	$16 \times 128 \times 128$
conv2	Convolution	$16 \times 120 \times 120$
activation_2	Activation	$16 \times 120 \times 120$
stoch_pooling_1	Stochastic pooling	$16 \times 60 \times 60$
dropout_1	Dropout	$16 \times 60 \times 60$
conv3	Convolution	$32 \times 56 \times 56$
activation_3	Activation	$32 \times 56 \times 56$
conv4	Convolution	$32 \times 52 \times 52$
activation_4	Activation	$32 \times 52 \times 52$
stoch_pooling_2	Stochastic pooling	$32 \times 26 \times 26$
dropout_2	Dropout	$32 \times 26 \times 26$
flatten_1	Flatten	$4608 \times 1 \times 1$
dense_1	Fully connected	$32 \times 1 \times 1$
activation_5	Activation	$32 \times 1 \times 1$
dropout_3	Dropout	$32 \times 1 \times 1$
dense_2	Fully connected	$28 \times 1 \times 1$
activation_6	Activation	$28 \times 1 \times 1$
Output	SoftMax regression	$28 \times 1 \times 1$

Let us assume an input video frame of size $I \in R^{w \times h}$. The convolutional kernel with size K is considered for convolution with a stride of S and P padding for filling the input video frame boundary. The size of the output of convolution layer is given by

$$S_{\text{OUT}} = \frac{(I - K + 2P)}{S} + 1. \tag{1}$$

The architecture of our CNN model consists of four convolutional layers. While the first two layers extract the low level features (like lines, corners, and edges), the last two layers learn high level features. The detailed layer information and their output sizes with parameters are tabulated in Table 1.

The output of a convolutional layer is generally denoted with the following standard equation as

$$y_j^n = f\left(\sum_{i \in c_j} y_i^{n-1} * k_{ij}^n + \zeta_j^n\right), \tag{2}$$

where n represents the nth layer, k_{ij} is the convolutional kernel, ζ_j represents bias, and the input maps are represented by c_j. The CNN uses a tanh activation function with an additive bias formulated as

$$\hbar_{ni}^{xy} = \tanh\left(\zeta_{ni} + \sum_{w=0}^{w_i - 1} \sum_{h=0}^{h_j - 1} W_{ij}^{wh} \hbar_{i-1}^{(x+w)(y+h)}\right). \tag{3}$$

ζ_{ni} represents feature map bias which are non-supervisory trained and w_i, h_j are the kernel width and height, respectively. W_{ij}^{wh} is the weight of the kernel at position (w, h). Over

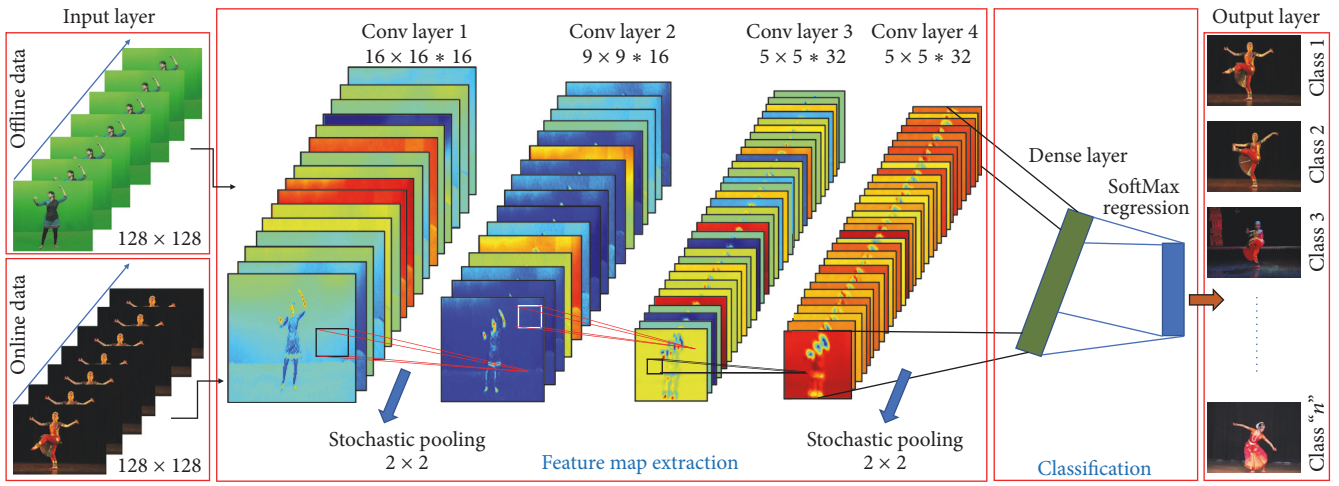


Figure 2: The proposed Deep CNN architecture.

a region, the max value of a feature is obtained using pooling technique, which reduces the data variance. We implemented our architecture with stochastic pooling technique by calculating the probability values for each region. For every feature map c, the probability is given by

$$\chi_{w,h}^{n,k} = \text{Stochastic}_{(w,h,i,j)\in p}\left(\chi_{w,h}^{n-1,k} u\left(i,j\right)\right), \tag{4}$$

where $\chi_{w,h}^{n,k}$ is the neuron activation function at a point (w,h) in spatial coordinates and $u(i,j)$ is the weighing function of window. When compared to other pooling techniques, stochastic pooling makes CNN converge at a faster rate and improves the ability of generalization in processing invariant features.

This Indian classical dance identification is a multiclass classification problem. Hence, a SoftMax regression layer given by a hypothesis function $h_\phi(x)$ is being used as

$$h_\phi\left(x\right) = \frac{1}{1 + e^{(-\phi^T x)}}. \tag{5}$$

ϕ must be trained in a way that the cost function $J(\phi)$ is to be minimized.

$$J\left(\phi\right) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=0}^{l} l\left\{y^i = j\right\}\log_p\left(y^i = \mathbb{Z} \mid x^i;\phi\right)\right]. \tag{6}$$

The classification probability in SoftMax regression layer for classifying an input x as a category $\mathbb{Z}$ is given as

$$p\left(y^i = \mathbb{Z} \mid x^i;\phi\right) = \frac{e^{\phi_j^T x^i}}{\sum_{l=1}^{k} e^{\phi_j^T x^i}}. \tag{7}$$

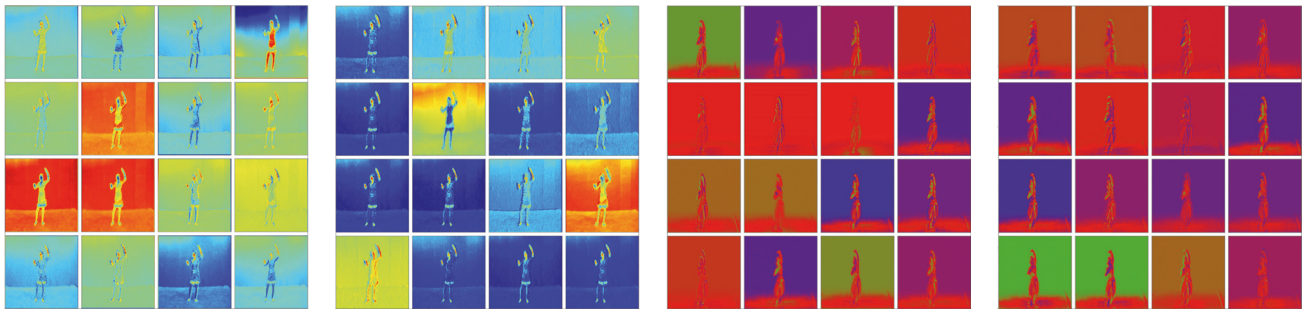
The network is trained to learn the features of each dance pose by means of a supervised learning. The internal feature representation reflects the likeness among training samples. We outline 200 poses from ICD (offline data) performed by 10 classical dancers. The size of the total dataset is 2000 dance poses with each pose recording normalized to 2 secs or 60 frames per second forming a total of 120k frames. Similarly, the online dance data is downloaded from YouTube and each pose normalized to 60 fps. All together to know the feature representation learned by the CNN system, the maximized activation neuron is extracted to recognize the dance pose accurately. Finally, the feature maps were visualized by averaging the image patches with stochastic response in higher layers.

4. Results and Discussion

The main goal of this work is to correctly identify the dance pose from an ICD dataset. We attempted Indian classical dance identification using SVM classifier [13], adaptive graph matching [17], traditional ANN [15], and deep ANN [16]. Among these classifiers, deep ANN outperformed them with massive recognition rates at a lower speed. But real-time implementation of ICD identification demands a high-speed classifier. To improve the speed of the recognition, Adaboost classifier [14] is introduced. Even though the recognition is fast, the classification results were found to be somewhat unreliable at times. Hence, this paper introduces using the powerful CNN tool to classify ICD poses at a faster speed with the best recognition rates.

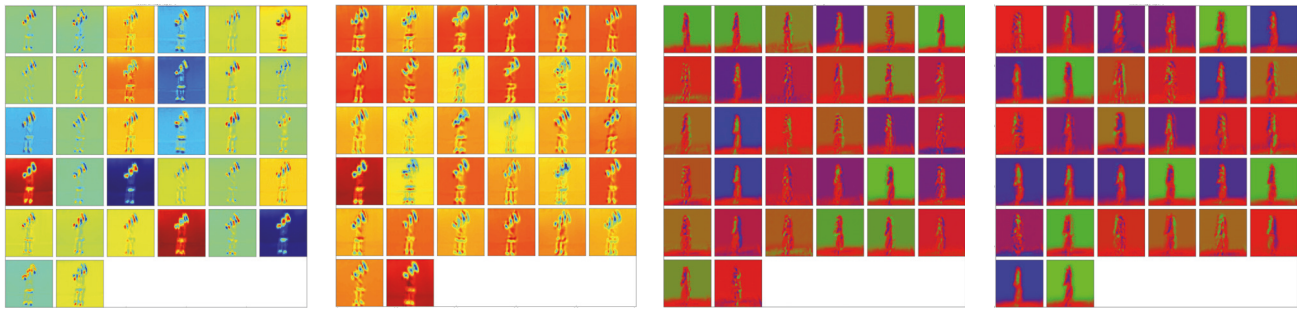
The proposed model of CNN is applied to the both online and offline Indian classical dance database for classification. The online database is downloaded from YouTube and the offline database is created in our laboratory as mentioned in Section 1. Each dance pose image in the dataset is preprocessed by reducing its dimensions to 128 × 128 × 3 which will improve the computational speed of CNN.

4.1. Batch I: CNN Training with Only One Set of Data. Training of our proposed CNN model is done in three batches. In Batch I of training, only one set of data, that is, 200 poses performed by one classical dancer captured with DSLR camera for 2 seconds at 30 fps forming a dataset with a total of 12000 images, is used. The images are preprocessed and training is initiated using our proposed CNN architecture.



(a) (b) (c) (d)

Figure 3: Feature maps. ((a) and (b)) Outputs of convolutional layers 1 and 2, respectively, for offline data. ((c) and (d)) Outputs of convolutional layers 1 and 2, respectively, for online data with 16 filters each.



(a) (b) (c) (d)

Figure 4: Feature maps. ((a) and (b)) Outputs of convolutional layers 3 and 4, respectively, for offline data. ((c) and (d)) Outputs of convolutional layers 3 and 4, respectively, for online data with 32 filters each.

Similarly, the online data is preprocessed and training is initiated. The CNN algorithm is implemented on Python 3.6 platform using a high-performance computing (HPC) machine with 6 CPU-GPU combination.

The CNN is trained using a gradient-descent algorithm at two stages. Stage one handles the multiclass classification problem with feedforward pass having S training samples from c classes. Stage two is the backpropagation pass. The error function is computed as

$$\varepsilon_e{}^S = \frac{1}{2}\sum_{m=1}^{S}\sum_{k=1}^{c}\left(l_k^m - v_k^m\right)^2, \tag{8}$$

where l_k^m is the label of mth pattern of kth dimension and v_k^m is the corresponding value of the layer unit. The output of the convolutional layer is the tanh activation function of this value. The backpropagation pass is from higher to lower layers and the error in nth layer is β_e^n calculated as

$$\beta_e^n = \left(w^{n+1}\right)^T \beta_e^{n+1} \odot f'\left(w^n y^{n-1} + \zeta^n\right). \tag{9}$$

The weight in nth layer is updated according to $\Delta w^n = -\lambda(\partial\varepsilon/\partial w^n)$.

During the separate training of online data and offline data, different feature maps were observed at different layers. Figures 3(a) and 3(b) visualize the feature maps of one offline dance pose frame obtained in convolutional layer 1 and convolutional layer 2 with 16 filters and Figures 3(c) and 3(d) for online dance pose frame.

Low level features like lines, edges, and corners are learned from convolutional layers 1 and 2. High level features learned from convolutional layers 3 and 4 are visualized in Figure 4. A stochastic pooling which combines the advantages of both mean and max pooling techniques is implemented. It also overcomes the problem of overfitting. Increasing the number of pooling layers will increase substantial information loss. Hence, the stochastic pooling is implemented in only two layers which is achieved by calculating the probability values of each region.

In Batch I we have used one dataset for training. Testing was carried out in two cases. In Case I of testing, the same dataset is used (i.e., training and testing were done on the same dataset); for Case II of testing, different dataset is used. In both the cases, good recognition rates were obtained and are tabulated in Table 2.

4.2. Batch II: CNN Training with Two Sets of Data. In this case, two sets of data created from two classical dancers are used for training. For this batch dataset is created with 200 Indian classical dance poses of five native classical dancers for 2 seconds each at 30 fps. Training is performed for two sets of data on HPC machine in 100 epochs. Testing is initiated in two cases as mentioned in the previous section. Case I of testing uses the same dataset which is used in

Table 2: Recognition rates in different CNN training cases with two testing conditions.

Training batch	Number of training datasets	Training datasets	Testing datasets	Recognition rates (%)	
				Offline data	Online data
Batch I (12000 images)	1	Dataset 1	Dataset 1	92.12	90.34
			Dataset 2	90.88	88.98
Batch II (60000 images)	5	Dataset 1 to Dataset 5	Dataset 1	92.83	91.52
			Dataset 6	91.17	89.87
Batch III (96000 images)	8	Dataset 1 to Dataset 8	Dataset 1	93.33	91.96
			Dataset 9	91.47	89.92
			Dataset 10	91.99	89.05

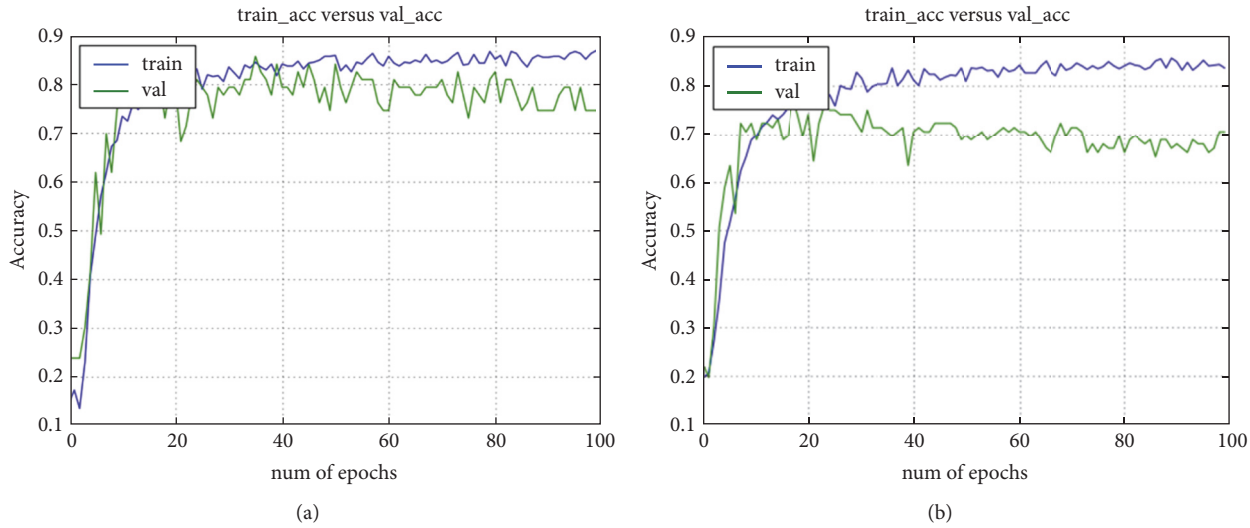


Figure 5: Training accuracy and validation accuracy (a) for offline data and (b) for online data.

training. For Case II of testing, the sixth dataset is used. The acquired recognition rates with both online and offline data were tabulated in Table 2. Here, by increasing the number of datasets for training it is observed that a good amount of recognition is achieved compared to Batch I training. It is also observed that the accuracy in recalling the pose is substantially increased as the number of training datasets increased. However, the training time increased by 50% than the Batch I training process.

4.3. Batch III: CNN Training with Three Sets of Data. Further improvement in recognition rates is achieved by increasing the training to CNN. A total of ten datasets were created, out of which 8 datasets were used in training and 2 sets for testing. An increase in recognition rates was obtained using this batch for training. Figure 5 shows the training accuracy versus validation accuracy plot for Batch III training set. It shows that the validation accuracy is good and with less amount of overfitting.

Figures 6(a) and 6(b) plot losses during training of Batch III on offline data and online data, respectively. There is a small difference in training and validation losses with an overall less than normal loss coefficient. An average confusion matrix is generated based on the recognition rates and the number of matches for Batch III of training and Class II of testing on both offline data and online data is shown in Figures 7(a) and 7(b), respectively. For better visualization, it is shown for only a limited number of ICD poses. Batch III with 8 multiple datasets of training is showing better recognition of signs compared to the other two batches. However, we sacrifice training computation time for recognition. Time of real-time recognition is 0.4 sec per frame and it is quite fast compared to algorithms like SVM and Fuzzy classifiers.

All convolutional layers are implemented with different filter windows of sizes 32×32, 16×16, 9×9, and 5×5. Reducing the filter size improves the recognition rates but increases the computational time due to the increase in number of filters. So, we used convolutional windows of sizes 16×16, 9×9, 5×5, and 5×5 for conv1, conv2, conv3, and conv4 layers, respectively. Table 3 compares the performance of choosing different filtering window sizes.

A stochastic pooling adoption attained an average recognition rate of 93.33%. Implementing max pooling and mean pooling produces a recognition rate of 91.33% and 89.84%, respectively.

To further know the robustness and efficiency of implementation of Indian classical dance identification with CNN,

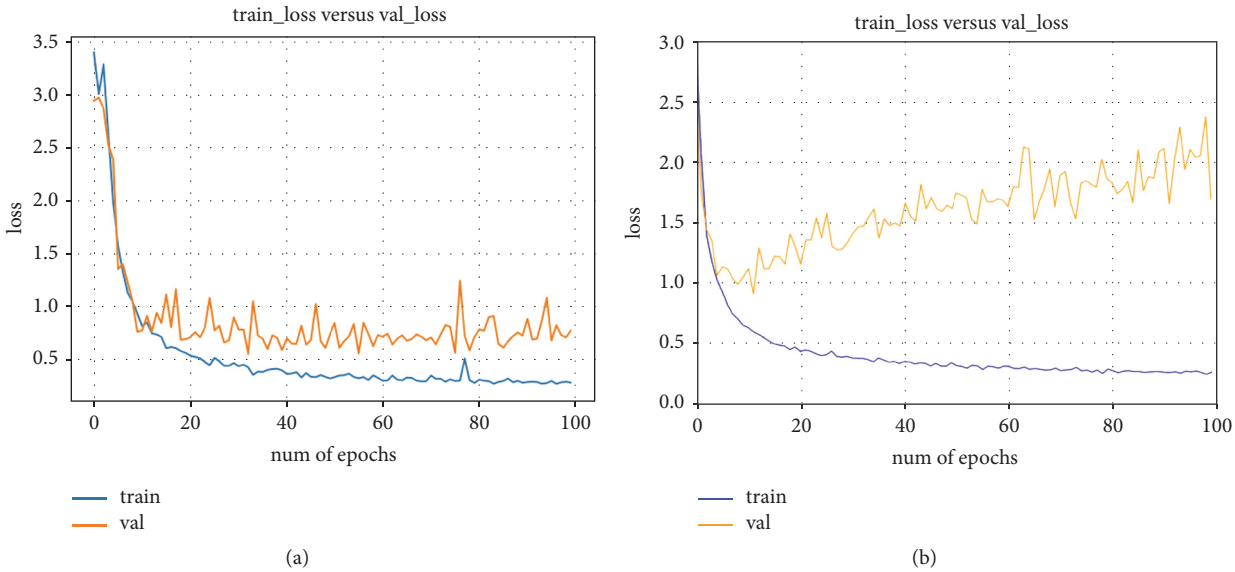


Figure 6: Training loss and validation loss (a) for offline data and (b) for online data.

Table 3: Performance comparison of CNN with different convolutional filter sizes.

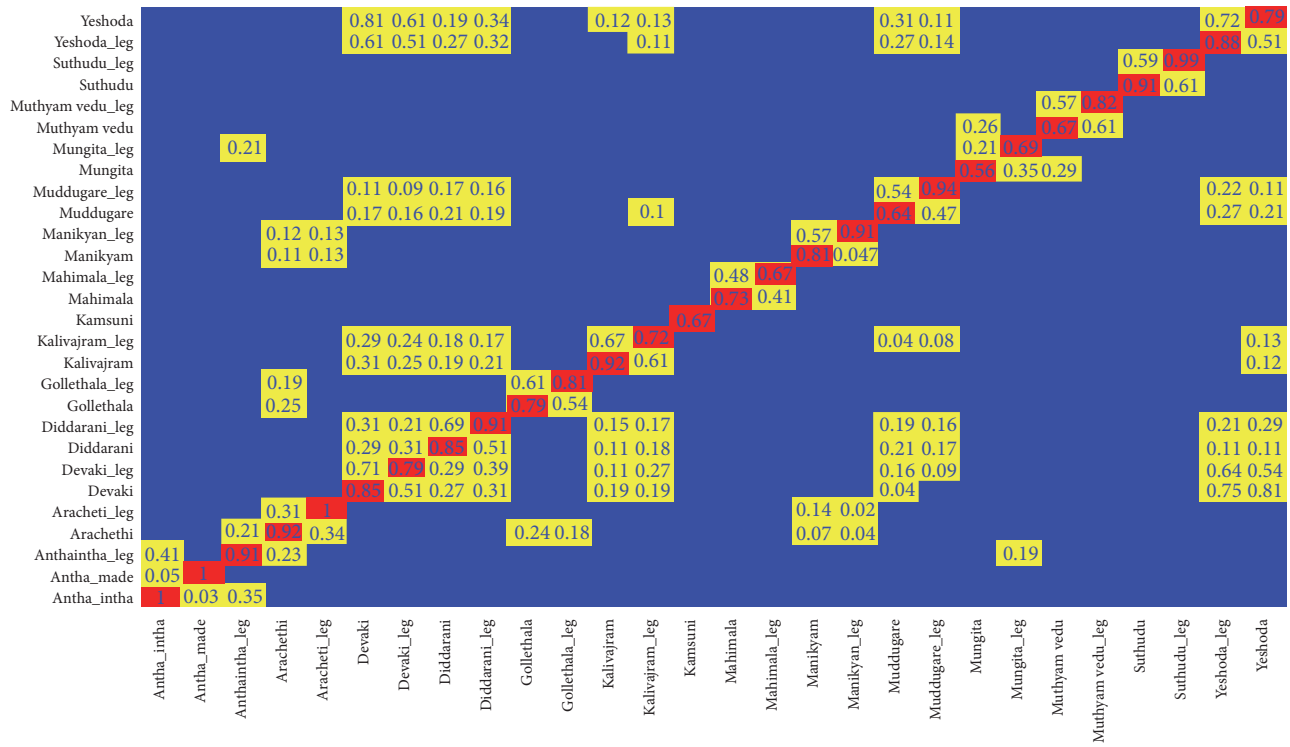
	Layers				Recognition rate (%)	Training times (in hours)
	conv1	conv2	conv3	conv4		
Convolutional filter window size	16 × 16	9 × 9	5 × 5	5 × 5	93.33	207
	5 × 5	5 × 5	5 × 5	5 × 5	95.54	296
	9 × 9	9 × 9	9 × 9	9 × 9	92.83	205
	16 × 16	16 × 16	16 × 16	16 × 16	90.15	168
	32 × 32	32 × 32	32 × 32	32 × 32	89.86	142

it is compared with other classifiers used in our previous works. For faster recognition, we used SVM in [13] and ended with very low classification rates. Further, we replaced SVM with Adaboost classifier [14] and found moderate recognition rates. Later, we used adaptive graph matching (AGM) model on the data and found good recognition rates. We also tried a traditional artificial neural network (ANN) which fails in producing good recognition rates. Further, good classification scores achieved using deep ANN for Indian classical dance identification. The novelty of our proposed CNN model is tested by comparing with other well-known CNN architectures LeNet [25] and VGG Net [27]. From Tables 4 and 5, it is observed that the proposed CNN architecture is promisingly working in identifying a correct label of a particular dance action pose.

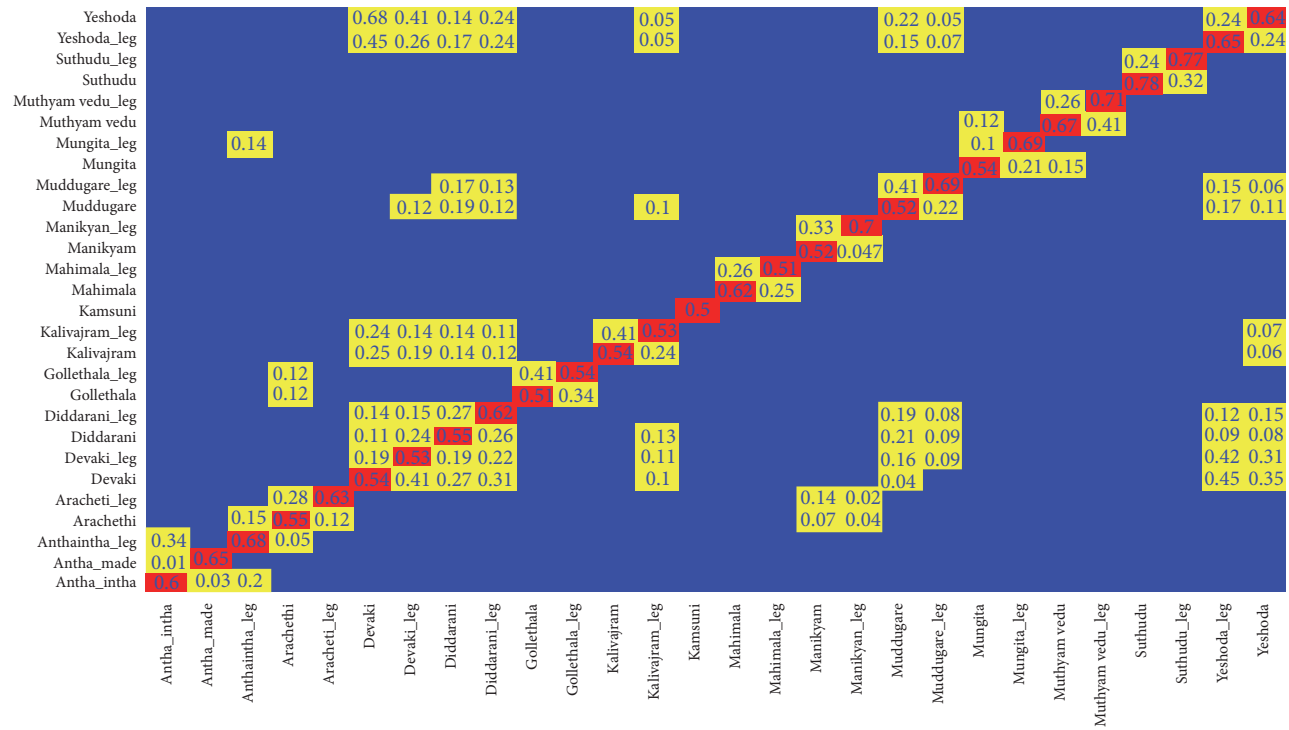
The recognition accuracy is further improved by replacing ANN with deep ANN and reported an increase in recognition rate by 5%. A much better improvement of 4% in the recognition accuracy and an upward 15% in testing speed were observed in this work with convolutional neural networks. Even though CNN takes more time for training, the testing takes a comparatively far lesser computation time. Recognition rates obtained with different classifiers for offline data and online data are compared in Tables 4 and 5, respectively. Hence, CNNs are a suitable tool for simulating Indian classical dance identification accurately. Testing is done on a 64-bit CPU with a 4 GB ram memory in Python 3.6 with OpenCV and Keras deep learning libraries.

5. Conclusion

CNN is a powerful artificial intelligence tool in pattern classification. In this paper, we proposed a CNN architecture for classifying Indian classical dance poses/mudras. The CNN architecture is designed with four convolutional layers. Each convolutional layer with different filtering window sizes is considered which improves the speed and accuracy in recognition. A stochastic pooling technique is implemented which combines the advantages of both max and mean pooling techniques. We created the offline classical dance database of 200 ICD poses with 10 classical dancers for 2 secs each at 30 fps generating a total of 120000 pose frames. Training is performed in different batches to know the robustness of enormous training modes required for CNNs. In Batch III of training, the training is performed with eight sets of data (i.e., 96000 video frames) and maximizing the recognition of the ICD. Training accuracy and validation accuracies for this CNN architecture are better than the previously proposed ICD classification models. A less amount of training and validation loss is observed with the proposed CNN architecture.



(a) Average confusion matrix generated in Batch III of training with Case II of testing for offline data



(b) Average confusion matrix generated in Batch III of training with Case II of testing for online data

FIGURE 7: Average confusion matrices generated for Indian classical dance poses.

Table 4: Recognition rates of offline dance data identification with different classifiers.

Classifier	Average recognition rates (%) of offline dance data					
	Batch I training		Batch II training		Batch III training	
	Testing with the same dataset	Testing with different dataset	Testing with the same dataset	Testing with different dataset	Testing with the same dataset	Testing with different dataset
ANN [15]	83.42	80.01	85.03	81.61	86.49	82.11
Deep ANN [16]	87.39	83.62	88.01	84.92	89.49	86.99
SVM [13]	75.93	71.31	78.35	74.48	80.46	76.78
Adaboost [14]	80.19	76.49	80.98	77.29	81.76	79.09
AGM [17]	87.20	83.16	87.89	84.63	88.23	85.05
LeNet [25]	88.14	85.49	88.55	86.32	87.92	86.85
VGG [27]	89.98	87.02	90.12	88.41	88.76	88.02
Our proposed CNN architecture	92.14	90.88	92.83	91.17	93.33	92.47

Table 5: Recognition rates of online dance data identification with different classifiers.

Classifier	Average recognition rates (%) of online dance data					
	Batch I training		Batch II training		Batch III training	
	Testing with the same dataset	Testing with different dataset	Testing with the same dataset	Testing with different dataset	Testing with the same dataset	Testing with different dataset
ANN [15]	82.82	79.13	84.16	80.46	85.95	81.19
Deep ANN [16]	85.27	81.26	86.45	82.91	87.43	84.87
SVM [13]	74.12	70.66	77.63	73.45	79.51	74.23
Adaboost [14]	80.01	74.54	79.49	76.62	80.55	78.10
AGM [17]	85.11	80.09	86.09	81.31	86.88	88.41
LeNet [25]	86.43	81.79	88.26	80.49	87.73	85.66
VGG [27]	88.52	84.54	89.15	85.45	88.19	86.76
Our proposed CNN architecture	90.34	88.98	91.52	89.87	91.96	89.92

The average recognition rate of the proposed CNN model is 93.33% and is higher compared with the other state-of-the-art classifiers.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] H. Rahmani, A. Mian, and M. Shah, "Learning a deep model for human action recognition from novel viewpoints," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1-1, 2017.

[2] E. Rodolà, S. R. Bulò, T. Windheuser, M. Vestner, and D. Cremers, "Dense non-rigid shape correspondence using random forests," in *Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014*, pp. 4177–4184, Columbus, OH, USA, June 2014.

[3] D. Das Dawn and S. H. Shaikh, "A comprehensive survey of human action recognition with spatio-temporal interest point (STIP) detector," *The Visual Computer*, vol. 32, no. 3, pp. 289–306, 2016.

[4] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *Proceedings of the 14th IEEE International Conference on Computer Vision (ICCV '13)*, pp. 3551–3558, Sydney, Australia, December 2013.

[5] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Action recognition by dense trajectories," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '11)*, pp. 3169–3176, June 2011.

[6] P. V. V. Kishore, M. V. D. Prasad, D. A. Kumar, and A. S. C. S. Sastry, "Optical Flow Hand Tracking and Active Contour Hand Shape Features for Continuous Sign Language Recognition with Artificial Neural Networks," in *Proceedings of the 6th International Advanced Computing Conference, IACC 2016*, pp. 346–351, Bhimavaram, India, February 2016.

[7] A. Jalal, Y. Kim, Y. Kim, S. Kamal, and D. Kim, "Robust human activity recognition from depth video using spatiotemporal multi-fused features," *Pattern Recognition*, vol. 61, pp. 295–308, 2017.

[8] S. Ji, W. Xu, M. Yang, and K. Yu, "3D Convolutional neural networks for human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, 2013.

[9] A. Mohanty, P. Vaishnavi, P. Jana et al., "Nrityabodha: towards understanding indian classical dance using a deep learning approach," *Signal Processing: Image Communication*, vol. 47, pp. 529–548, 2016.

[10] Y. Yang and D. Ramanan, "Articulated pose estimation with flexible mixtures-of-parts," in *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011*, pp. 1385–1392, Colorado Springs, Colo, USA, June 2011.

[11] F. Wang and Y. Li, "Beyond physical connections: tree models in human pose estimation," in *Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2013*, pp. 596–603, Portland, Ore, USA, June 2013.

[12] S. Samanta, P. Purkait, and B. Chanda, "Indian Classical Dance classification by learning dance pose bases," in *Proceedings of the 2012 IEEE Workshop on the Applications of Computer Vision, WACV 2012*, pp. 265–270, Breckenridge, Colo, USA, January 2012.

[13] K. V. Kumar and P. V. Kishore, "Indian Classical Dance Mudra Classification Using HOG Features and SVM Classifier," in *Smart Computing and Informatics*, vol. 77 of *Smart Innovation, Systems and Technologies*, pp. 659–668, Springer Singapore, Singapore, 2018.

[14] K. V. Kumar, P. V. Kishore, and D. Anil Kumar, "Indian Classical Dance Classification with Adaboost Multiclass Classifier on Multifeature Fusion," *Mathematical Problems in Engineering*, vol. 2017, pp. 1–18, 2017.

[15] G. A. Rao, P. V. V. Kishore, D. A. Kumar, and A. S. C. S. Sastry, "Neural network classifier for continuous sign language recognition with selfie video," *Far East Journal of Electronics and Communications*, vol. 17, no. 1, pp. 49–71, 2017.

[16] G. A. Rao and P. V. V. Kishore, "Selfie video based continuous Indian sign language recognition system," *Ain Shams Engineering Journal*, 2016.

[17] X. Yang, H. Qiao, and Z.-Y. Liu, "Point correspondence by a new third order graph matching algorithm," *Pattern Recognition*, vol. 65, pp. 108–118, 2017.

[18] Z. Dong and X. Tian, "Multi-level photo quality assessment with multi-view features," *Neurocomputing*, vol. 168, pp. 308–319, 2015.

[19] Z. Dong, X. Shen, H. Li, and X. Tian, "Photo quality assessment with DCNN that understands image well," in *Proceedings of the International Conference on MultiMedia Modeling (MMM)*, Lecture Notes in Computer Science, pp. 524–535, 2015.

[20] X. Lu, Z. Lin, H. Jin, J. Yang, and J. Z. Wang, "Rapid: Rating pictorial aesthetics using deep learning," in *Proceedings of the 2014 ACM Conference on Multimedia, MM 2014*, pp. 457–466, USA, November 2014.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS '12)*, pp. 1097–1105, Lake Tahoe, Nev, USA, December 2012.

[22] Y. Sun, X. Wang, and X. Tang, "Deep learning face representation from predicting 10,000 classes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '14)*, pp. 1891–1898, Columbus, Ohio, USA, June 2014.

[23] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?" in *Proceedings of IEEE 12th International Conference on Computer Vision (ICCV '09)*, pp. 2146–2153, Kyoto, Japan, October 2009.

[24] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proceedings of the 26th International Conference on Machine Learning (ICML '09)*, pp. 609–616, Montreal, Canada, June 2009.

[25] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[26] J. Deng, W. Dong, and R. Socher, "ImageNet: a large-scale hierarchical image database," in *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255, Miami, Fla, USA, June 2009.

[27] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of the in International Conference on Learning Representations (ICLR*, 2015.

[28] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–27, 2009.

[29] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998.

[30] H. Lee, A. Battle, R. Raina, and A. Y. Ng, "Efficient sparse coding algorithms," in *Proceedings of the 20th Annual Conference on Neural Information Processing Systems (NIPS '06)*, pp. 801–808, December 2006.

[31] R. Salakhutdinov and G. E. Hinton, "Deep boltzmann machines," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, vol. 24, pp. 448–455, Clearwater Beach, Florida USA, 2009.