



ORIGINAL

Posture Recognition in Bharathanatyam Images using 2D-CNN

Reconocimiento de posturas en imágenes Bharathanatyam mediante 2D-CNN

M. Kalaimani¹  , AN. Sigappi¹  

¹Department of Computer Science & Engineering. Annamalai University. Annamalai Nagar, Chidambaram, India

Cite as: Kalaimani M, Sigappi A. Posture Recognition in Bharathanatyam Images using 2D-CNN. Data and Metadata 2023; 2:136. <https://doi.org/10.56294/dm2023136>.

Submitted: 07-09-2023

Revised: 12-11-2023

Accepted: 03-12-2023

Published: 04-12-2023

Editor: Prof. Dr. Javier González Argote 

ABSTRACT

The postures are important for conveying emotions, expressing artistic intent, and preserving appropriate technique. Posture recognition in dance is essential for several reasons, as it improving the performance and overall artistic expression of the dancer. The Samapadam, Aramandi, and Muzhumandi are three postures that serve as the foundation for the Bharathanatyam dance style. This work proposes a model designed to recognize the posture portrayed by the dancer. The proposed methodology employs the pre-trained 2D-CNN model fine-tuned using the Bharathanatyam dance image dataset and evaluates the model performance.

Keywords: Bharathanatyam Postures; 2D-CNN; Evaluation Metrics.

RESUMEN

Las posturas son importantes para transmitir emociones, expresar intenciones artísticas y conservar una técnica adecuada. El reconocimiento de las posturas en la danza es esencial por varias razones, ya que mejora el rendimiento y la expresión artística general del bailarín. El Samapadam, el Aramandi y el Muzhumandi son tres posturas que sirven de base al estilo de danza Bharathanatyam. Este trabajo propone un modelo diseñado para reconocer la postura representada por la bailarina. La metodología propuesta emplea el modelo 2D-CNN pre-entrenado y afinado utilizando el conjunto de datos de imágenes de danza Bharathanatyam y evalúa el rendimiento del modelo.

Palabras clave: Posturas Bharathanatyam; 2D-CNN; Métricas De Evaluación.




INTRODUCTION

Posture refers to the position and alignment of various parts of the body and a proper posture involves maintaining a balanced and neutral alignment of the body that allows for optimal support, stability and function. Postures in dance refer to the specific positions and alignments of the body that dancers assume while performing various movements and routines.

Bharathanatyam is a classical Indian dance form that traces its root to the state of Tamil Nadu. Developing and mastering the correct Bharathanatyam posture requires patience, hard work, practice, and is essential for adding refinement to the dance form.⁽¹⁾ There are basically three types of dance postures which are actually performed with hand mudras, eye movements and expressions. These postures are the Samapadam, Aramandi, and the Muzhumandi are shown in Table 1.

Posture recognition refers to the ability to identify and analyze the position and movements of a person's body, usually using technology such as cameras, sensors, or machine learning algorithms. The reasons to recognize posture in dance are technique improvement, injury prevention, aesthetic quality, expressive communication, feedback and coaching, consistency and progress tracking, skill enhancement and advancing the overall quality of dance performances.

Table 1. Basic postures of Bharathanatyam dance

S.NO	Posture Name	Posture	Description
1.	Samapadam		The Samapadam (or sthanakam) is the most basic form of Bharathanatyam postures. ⁽¹⁾ In this posture, the body is maintained upright with the knees slightly bent to achieve a balanced posture. The feet are aligned without any space.
2.	Aramandi		This posture is most commonly used. Aramandi (Ardhamandalam) or Ayatham can be done by making the body third-quarter and turning the feet sideways to stabilize the body.
3.	Muzhumandi		Muzhumandi position is the most complete sitting position, where the feet position is maintained in the same as the Aramandi. In this position, thighs are stretched in the opposite direction, heels are held in a joined position, and feet are kept apart to balance the body.

Literature survey

Xiaodan Hu et al.⁽²⁾ developed a Hierarchical Dance video recognition model framework for extracting body motion and dance genres. The framework was used to create a new University of Illinois Dance Dataset contain 1143 video clips. The LSTM network was used for 3D movement subsequence extraction for dance genre recognition, receiving an F- score of 86 %.

Using a Kinect sensor to capture depth images, Wen June Wang et al.⁽³⁾ were able to categorize the 5 different postures of human (Sitting, Standing, Stooping, Kneeling, and Lying) using a LVQ neural network. They classified postures using LVQ and identified Horizontal projection, Vertical projection, and Star Skeleton for body position information. Their proposed posture recognition algorithm outperforms the average by 99 %.

Matthias Dantone et al.⁽⁴⁾ relied on still images to predict 2D human poses. Specifically, they used 2-layer random forests as common denominators. The discriminant independent classifier for body parts acted as a first layer. The next layer takes predictable class distributions and predicts joint locations. They used 'Leeds

Sports Pose' dataset and the newly collected 'FashionPose' dataset with a high degree of fabric and appearance variation. They achieved 55,5 % accuracy, higher than previous SVM results

In this study, Rinki Gupta et al.⁽⁵⁾ used a wearable sensor to detect normal or abnormal human body posture using 3 tri-axis accelerometers placed at the backs of seven volunteers. This study used nine different sitting or standing postures. The experimental data showed that the HDN with the LSTM had the accuracy (99,91 %) compared to the HDN alone (97,88 %) and the LSTM alone (88,47 %). Complexity and execution time were also compared between the models.

Lei Zhao⁽⁶⁾ distinguished four basketball postures: dribbling, passing, catching, and shooting. Four nine-axis inertial sensors worn on the arm captured the data. The Support Vector Machine (SVM) approach was used to recognize posture after 30 eigenvectors were produced by Principal Component Analysis (PCA). They compared the Back Propagation Neural Network (BPNN) accuracy of 85,4 % to the SVM accuracy of 96 %.

Weili Ding⁽⁷⁾ described a method for recognizing human posture using Kinect sensor by deriving skeleton information. Multiple features, including angle and distance between joints, were defined. Then, using bagging and random subspace approaches, rule ensembles based on the RIPPER rule learning algorithm were created. They have used three datasets namely MSR-Action3D, Microsoft MSRC-12, and UTKinect-Action, and a fourth dataset "Baduanjin posture" was created using the Kinect sensor. They attained accuracy rates of 94,5 %, 97,6 %, 98,1 %, and 99,6 %, respectively.

Sriparna Saha⁽⁸⁾ identified 20 ballet dance positions. The skin color segmentation was used, then diluted and processed to produce skeletons of the postures. Staggered stick figure diagrams were transubstantiated in order to confirm the affirming reduced skeletons. The 20 postures were divided into five categories according to their Euler number and an empirically determined threshold was used to determine the accuracy of 91,35 %.

Gautam⁽⁹⁾ proposed a system for the recognition of 50 postures taken from 9 distinct Bharathanatyam videos. The HOG (histogram oriented gradient) features were extracted to categorize the postures. A Support Vector Machine (SVM) was implemented to identify the pose. After experimenting with different cell sizes, it was concluded that SVM was most accurate with a cell size [8,8] of 99,39 %.

Shubhangi⁽¹⁰⁾ work was to identify poses in Bharathanatyam, Kathak or Odissi. 15 poses were taken with a total of 100 images. Hu moments were used for describing the shape of an image, as they are scalar, translatable and rotation-independent. The resulting images were converted to binary form and SVM, using both the 1-1 and 1-N approaches. The accuracy was estimated to be 90 % and 83,33 %.

Tanwi Mallick⁽¹¹⁾ used nuiCapture software to create a Dataset of 10 sets of 58 ADAVUS by 9 dancers with the use of a Kinect XBox 360. The Key Postures were extracted from the video using audio beats as well as no-motion information. Features were extracted from both the skeleton and RGB images of the Kinect, which were then used to create GMM classifiers, SVM classifiers, and CNN classifiers. The accuracy of the posture recognition was reported to be 83,04 %; 97,97 %; and 61,30 %.

Creating a posture recognition system can be a challenging task, as it involves various technical, data-related, and practical challenges. Posture recognition systems may need to work in different environments with varying lighting conditions, backgrounds, and noise. Ensuring the model's robustness to these factors can be challenging. This work proposes to addressing the challenges by creating an extensive collection of images and developing a model to precisely recognize the postures.

METHODS

The proposed work on posture recognition comprises of input image acquisition, splitting the image dataset into training, validation and test datasets, creating and fine tuning model using 2D-CNN and test the model for performance assessment as shown in Figure 1.

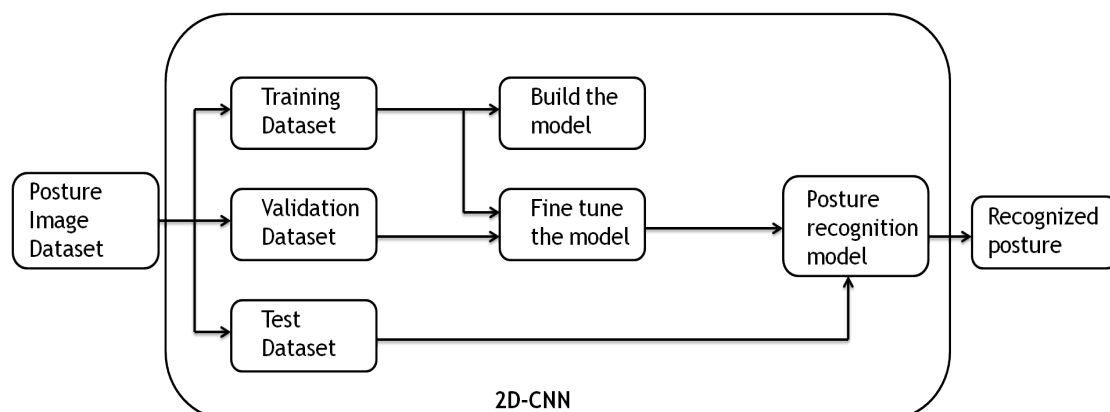


Figure 1. Proposed Model using 2D-CNN

Dataset creation

The posture image dataset is created using by capturing images of Bharathanatyam dancers portraying the three postures with Canon EOS 760D model camera. 600 images with resolution 4000x6000 are captured for each posture and the dataset has a total of 1800 images.

These images are taken by including the varying backgrounds, lightings and multiple perspectives of same posture to ensure the model can recognize the posture from an image captured in any manner. The images of dancers include a wide variety of costumes and accessories and with or without the artistic makeup of typical Bharathanatyam dancers. Also the dataset includes the dancers of various age groups, different skin colour and gender. Sample images of Samapadam posture taken in various attire against different background and various perspectives are shown in Figure 2.



Figure 2. Sample images of class Samapadam

Model implementation

The key components of a 2D-CNN include convolutional layers, pooling layers, fully connected layers, and an output layer. A series of convolutional layers followed by a pooling layer followed by the same iteration twice in 2D-CNN architecture. Two fully connected layers are then used to separate the poses after flattening the features to one dimension. Figure 3 shows a similar situation.

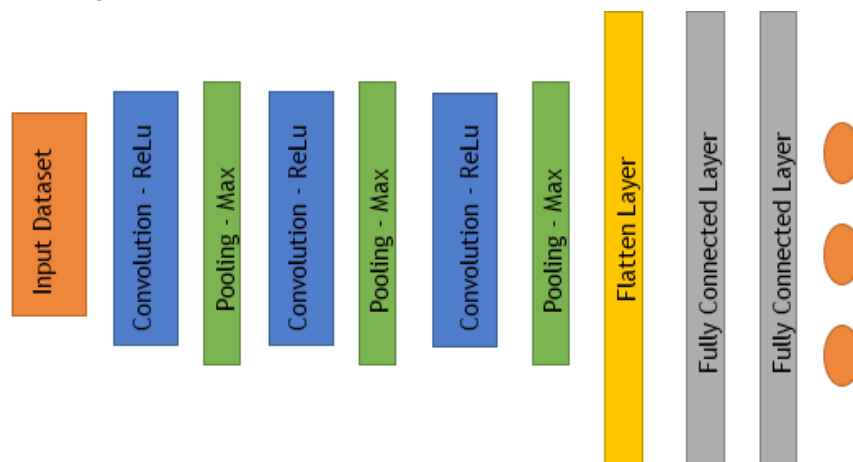


Figure 3. Architecture of 2D-CNN

Input layer

The dataset of 1800 images are used and the images are resized to 224x224 pixels. Each class of 600 images is split into 420 images for training the model and 90 images for model validation and 90 images to test the fine-tuned model. Training images and validation images are augmented in the proposed model.

Convolution layers

A convolutional layer is developed primarily for processing web-like inputs such as images and videos. Convolution operations are used in the convolution layer to extract features from input images. The layer by layer explanation of 2D-CNN used is given below.

- Input Layer: The resized images of size 224x224 is given as input.
- Convolutional layer 1: The convolutional layer is implemented with 16 kernels with size 2x2. These kernels will slide over the input image to perform convolution operations or to detect a specific pattern or feature in the input data. Padding determines how the convolution operation handles the edges and to ensure the edge information are not lost. “ReLU” activation function, which is widely used for introducing non-linearity and mitigate vanishing gradient problem is used.
 - Pooling Layer 1: This layer is often used in CNN to reduce the spatial dimensions and extract important information. Here, Max pooling layer is applied with size of pooling window 2x2. This layer divides the input feature map into non-overlapping 2x2 regions and selects the maximum value from each region to create a new down-sampled feature map. So, it effectively reduces the spatial dimensions of the feature map by a factor of 2. Now, the feature map of size 112x112 is passed to the next layer.
- Convolutional layer 2: This layer is implemented with 32 kernels with size 2x2 to perform the convolutional operations.
 - Pooling layer 2: This layer is implemented with Max pooling of size 2x2. Again this will reduce the feature map by half. Now, the feature map of size 56x56 is passed to next layer.
- Convolutional layer 3: This layer is implemented with 64 kernels with size 2x2 perform the convolutional operations.
 - Pooling layer 3: This layer is implemented with Max pooling with size 2x2. Again this will reduce the feature map by half. Now, the feature map of 28x28 is passed to next layer.
- Flatten layer: This layer converts the 2D feature map into 1D by arranging them sequentially in a single line. So, 28x28 feature maps with 64 kernels are converted to $28 \times 28 \times 64 = 50176$ neuron values.
- Fully connected layer 1: This layer is implemented with ReLU activation function, which introduces non-linearity by passing positive values and zero's. Also, this layer reduces the feature map size into 500. The purpose of this layer is to learn complex combinations of features from the previous layers. It can capture high-level patterns and relationships between features.
 - Fully connected layer 2: This further refine the features learned by the first fully connected layer. It has its own set of weights and biases and can capture even more complex patterns and relationships in the data. This layer applies softmax activation function and reduces the feature map size to 3, as the proposed model is a 3 class recognition system.
- Output layer: It predicts the category of the posture which has the highest weight.

RESULTS AND DISCUSSION

A 2D-CNN model summary gives a brief description of the network design, including details on the number of trainable parameters, the shapes and sizes of the layers. It is an effective tool for comprehending the intricacy and makeup of the neural network. Here's is the method to calculate total trainable parameters from them:

- For Convolutional Layers: The trainable parameters associated with the layer's filters and biases are calculated as: $[(\text{Height}_{\text{kernel}} * \text{Weight}_{\text{kernel}} * \text{No of kernel}_{\text{previous layer}}) + 1] * \text{No of kernel}_{\text{current layer}}$
- For Fully Connected (Dense) Layers: The trainable parameters can be calculated as: $(\text{No of neurons}_{\text{previous layer}} * \text{No of neurons}_{\text{current layer}}) + \text{No of neurons}_{\text{current layer}}$
- Total Trainable Parameters: Summing up the parameters from all convolutional and fully connected layers in network.

Hence, the calculation of trainable parameters on each layer for a posture recognition model is as follows:

- For convolution layer 1:
Number of trainable parameters = $[(2 * 2 * 3) + 1] * 16 = 208$
- For convolution layer 2
Number of trainable parameters = $[(2 * 2 * 16) + 1] * 32 = 2080$
- For convolution layer 3
Number of trainable parameters = $[(2 * 2 * 32) + 1] * 64 = 8256$
- For Fully connected layer 1

- Number of trainable parameters = $(50176 * 500) + 500 = 25,088,500$
- For Fully connected layer 2
Number of trainable parameters = $(500 * 3) + 3 = 1,503$
- Total trainable parameters = $208 + 2080 + 8256 + 25,088,500 + 1503 = 25,100,547$.

The snapshot of the model summary of the 2D-CNN model developed for posture recognition is given in Figure 4.

```

Model: "sequential"
-----
Layer (type)                Output Shape                Param #
-----
conv2d (Conv2D)             (None, 224, 224, 16)       208
max_pooling2d (MaxPooling2D) (None, 112, 112, 16)       0
conv2d_1 (Conv2D)           (None, 112, 112, 32)       2080
max_pooling2d_1 (MaxPooling2D) (None, 56, 56, 32)         0
conv2d_2 (Conv2D)           (None, 56, 56, 64)         8256
max_pooling2d_2 (MaxPooling2D) (None, 28, 28, 64)         0
flatten (Flatten)           (None, 50176)              0
dense (Dense)                (None, 500)                25088500
dense_1 (Dense)              (None, 3)                  1503
-----
Total params: 25,100,547
Trainable params: 25,100,547
Non-trainable params: 0
    
```

Figure 4. Model summary of 2D-CNN

Figure 5 shows the accuracy and loss graph of the proposed model. An accuracy and loss graph is a visualization of the performance of the CNN during training and/or validation. These graphs offer important information about the model's ability to generalize to new data and how well it is learning from the training data.

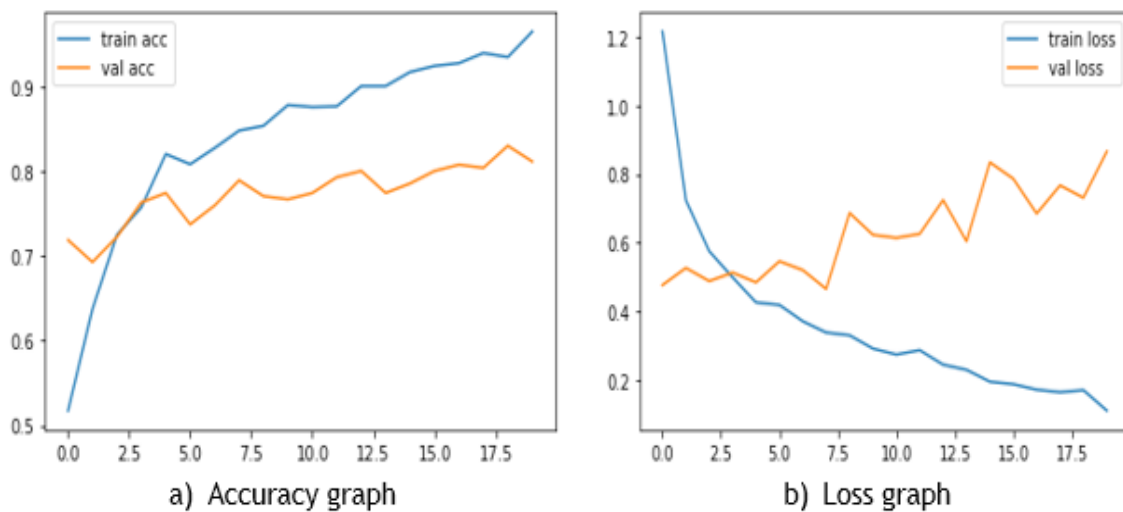


Figure 5. Results obtained from posture recognition system

Evaluation metrics

A confusion matrix can be used to evaluate the performance of a classification model. It provides an extensive evaluation of the model's predictions and their agreement with the actual labels. A confusion matrix provides the information about True positives, True negatives, False positives and False negatives.^(11,12,13)

		Predicted Class			Total
		Samapadam	Aramandi	Muzhumandi	
Actual Class	Postures				
	Samapadam	175	5	0	180
	Aramandi	8	172	0	180
	Muzhumandi	0	1	179	180
Total		183	178	179	540

Figure 6. Confusion matrix for posture recognition

Figure 6 shows the confusion matrix of posture recognition system. It is observed that the class Muzhumandi is better recognized than other postures. A confusion matrix provides a more detailed and informative view of a 2D CNN's performance. Accuracy, precision, Recall and F1-score are measures often used to assess the performance of models. It helps to understand the types of errors the model is making and guides in fine-tuning the model. With help of the formulas, the performance metrics are calculated and listed in table 2.

Performance Measure	Result (%)
Accuracy	97,4
Precision	97,2
Recall	97,4
F1 Score	97,4

An accuracy of 97,4 % means that the model correctly predicted the class labels for 97,4 % of the samples. This is an impressive accuracy and suggests that the proposed model makes accurate predictions in most cases. 97,2 % of precision means that, if the model predicts a positive class, it is correct 97,2 % of the time. A recall of 97,4 % indicates that the proposed model captures 97,4 % of all true positive cases in the dataset.

CONCLUSIONS

Posture recognition technology using image processing is a game-changer in enhancing dance performance. This work introduces a specialized model employing pre-trained 2D-CNN, fine-tuned with a dedicated Bharathanatyam dataset, to accurately identify these crucial postures. The evaluation of this model incorporates key metrics like accuracy, precision, recall, and F1-score, providing a comprehensive assessment of its effectiveness in the context of dance posture recognition. Such advancements in technology and methodology contribute to the preservation and advancement of this intricate art form, ensuring the continued excellence of dancers and the meaningful communication of emotions and artistic intent.

BIBLIOGRAPHIC REFERENCES

1. Podium School. Bharatanatyam postures for beginners. Disponible en: <https://learn.podium.school/bharatanatyam/bharatanatyam-postures-for-beginners/>
2. Hu X, Ahuja N. Unsupervised 3D Pose Estimation for Hierarchical Dance Video Recognition. En: 2021 IEEE/CVF International Conference on Computer Vision (ICCV).
3. Wang WJ, Chang JW, Haung SF, Wang RJ. Human Posture Recognition Based on Images Captured by the Kinect Sensor. *Int J Adv Robot Syst.* 2015.
4. Horta-Martínez LE. 3D printing in the medical field. *Seminars in Medical Writing and Education* 2022;1:8-8. <https://doi.org/10.56294/mw20228>.
5. Dantone M, Gall J, Leistner C, Van Gool L. Human Pose Estimation using Body Parts Dependent Joint Regressors. En: *IEEE Conference on Computer Vision and Pattern Recognition.* 2013.
6. Gupta R, Saini D, Mishra S. Posture detection using Deep Learning for Time Series Data. En: *Proceedings of the Third International Conference on Smart Systems and Inventive Technology.* 2020.
7. Zhao L, Chen W. Detection and Recognition of Human Body Posture in Motion Based on Sensor Technology. *IEEJ Trans Electr Electron Eng.* 2019.
8. Ding W, Hu B, Liu H, Wang X, Huang X. Human posture recognition based on multiple features and rule learning. *Int J Mach Learn Cybern.* 2020.
9. Saha S, Konar A. Topomorphological approach to automatic posture recognition in ballet dance. *IET Image Process.* 2015.
10. Gautam S, Joshi G, Garg N. Classification of Indian Classical Dance Steps using HOG Features. *Int J Adv Res Sci Eng.* 2017;6(8).
11. Shubhangi, Tiwary US. *Classification of Indian Classical Dance Forms.* Springer International Publishing. 2017.
12. Canova-Barrios C, Machuca-Contreras F. Interoperability standards in Health Information Systems: systematic review. *Seminars in Medical Writing and Education* 2022;1:7-7. <https://doi.org/10.56294/mw20227>.
13. Mallick T, Das PP, Majumdar AK. Posture and Sequence Recognition For Bharathanatyam dance Performances Using Machine Learning Approach. *J Vis Commun Image Represent.* 2022;87.

FINANCING

None

CONFLICT OF INTEREST

The authors declare that there is no conflict of interest.

AUTHORSHIP CONTRIBUTION

Conceptualization: M. Kalaimani

Data curation: M. Kalaimani

Formal analysis: M. Kalaimani

Acquisition of funds: NONE

Research: M. Kalaimani, Dr. AN. Sigappi

Methodology: M. Kalaimani

Project management: M. Kalaimani, Dr. AN. Sigappi

Resources: M. Kalaimani

Software: M. Kalaimani

Supervision: M. Kalaimani, Dr. AN. Sigappi

Validation: M. Kalaimani

Display: M. Kalaimani, Dr. AN. Sigappi

Drafting - original draft: M. Kalaimani

Writing - proofreading and editing: M. Kalaimani, Dr. AN. Sigappi