AUTOMATIC ANNOTATION OF DANCE VIDEOS BASED ON FOOT POSTURES

Shailesh S.
Research Scholar
Department of Computer Applications
Cochin University of Science and Technology
Kalamaserry, Kochi, Kerala, India

Dr. Judy M V
Associate Professor
Department of Computer Applications
Cochin University of Science and Technology
Kalamaserry, Kochi, Kerala, India

Abstract - Along with the advancements in the field of artificial intelligence, machine learning, and deep learning video annotation has become a more exciting and crucial problem, especially the video from medical, arts performance games and so on. In India, there is an increased demand for digitizing and archiving of arts and culture . This paper focuses on annotating dance videos based on foot postures (stanas) in an automatic manner. Using transfer-learning, the features from the images are extracted and a deep neural network is used for image classification. A trained model of Deep Stana Classifier is used for identifying stanas from the frames of a video. A complete annotation system comprises of a coupled architecture, one Deep Stana Classifier module, and an annotation module. The findings on the accuracy obtained for the Deep Stana Classifier produced positive results. In the second phase, the result of the annotations made on the video is kept as an index structure in JSON object format. The scope and opportunity of this work in the future is consistent and useful for annotating dance videos as well as videos from another domain.

Keywords: *dance, stanas, video-annotation, cnn, transfer learning, deep learning, dnn.*

1. Introduction

The existence of multimedia content of dance videos in formats like video and audio recordings opens up vast opportunities for computational analysis. In the field of classical dance, particularly South Indian Classical Dance (SICD), the knowledge transfer from gurus to the disciples is verbal. Some notations to analyze human movement like Labanotation and Benesh are used to represent dance as a sequence of symbols; still, it is insufficient to depict for the depth of South Indian classical dance forms like Bharatanatyam, Kuchipudi, Mohiniyattam, and Kathakali because these are rich with its theoretical constructs and aesthetic perfections.. Research on these classical dances, including constructs like stanas, hastabedas, karanas, etc., is supported by various governmental and non-governmental institutions but the information in this domain is scattered across different books, expert personalities, photographs, videotapes etc. The lack of linkage between this information has led to the difficulty in gathering resources in research due to the lack of ICT support in this domain.

For the digitization and archiving of dance resources, preparing videos is a cumbersome task. This paper proposes a method for automatic annotation of videos based on the foot work or stanas. Different methods like deep learning for classification , image processing for preprocessing, and computer vision for analysis were used while implementing this system. The challenges of automatic annotation can be divided into two levels. The first level is to build a classifier that can identify different stanas based on the postures present in the video frame. The second level is to process the video as a sequence of images and apply classification for identifying the stanas after which a meta-annotation file will be generated which represents the presence of a particular stana along with the timestamp value. With higher accuracy and efficiency, the classifier outperformed the traditional machine learning technique for dance stana classification.

2. Literature Review

Research on automatic annotation of dance videos is limited especially when pertaining to Indian classical dance forms. This literature review looks at similar works done on dance forms from other nations.

Huma Chaudhry et. al. [1]proposes a solution for complex dance motion annotation by an automated tool based on a dataset of different types of Malaysian dances available on YouTube. Ramadoss and Rajkumar [2]proposes a dance video semantics model where objects in a dance video, at multiple granularity levels, are identified by the accompanying song components with a new entity type, Agent, to represent the spatio-temporal actions of actors. The authors also introduce a conceptual multimedia data model called Video Semantics Directed Acyclic Graph (VSDAG) based on protocol analysis and is stored in an XML compliant structure. Balakrishnan Ramadoss [3] introduces an architecture of manual annotation and semi-automatic authoring tool with a search engine called DMAR (Dance Media Annotation, Authoring, and Retrieval System) to demonstrate dance media semantic annotation. It also describes the XML schema-based content description structure of DMAR and proposes a quality metric fidelity to evaluate the annotations. Balakrishnan Ramadoss[4] proposes a new meta-model, Dance Video Content Model (DVCM) to identify the expressive video semantics at multiple granularity levels by a new relationship type called Temporal Semantic Relationship to reduce the search time of dance queries and create an inverted file-based index. Lei Chen[5] presents a model for expressing the semantics of video data by the relationship between video data and DISIMA images made through keyframes as well as a set of new predicates to describe the spatio-temporal characteristics of salient objects in the video data by using MOQL query language. A.Ekin [6]developed a new integrated semantic-syntactic video model with semantic entities. The relations between these entities are devised for structured video search and browsing, introducing a new entity called 'actor' for grouping of objects. As a conclusion, there were no methods proposed for annotating classical dance videos based on stanas.

3. Preliminaries

3.1. Convolution Neural Networks (CNN) and Transfer Learning

Many computer vision methods are based on a Convolutional Neural Network (CNN)[7]. In comparison to conventional multilayer perceptron architectures, it uses two operations called 'convolution' and 'pooling' to reduce an image to its essential features, using those features to recognize and identify the picture. In Figure 1[7], a general architecture of a Convolutional Neural Network (CNN) is shown. The central building squares of CNN are Convolution layer, activation layer, pooling layer and fully connected layer. Convolution layer is a "filter", sometimes called a 'kernel', which passes over a picture, operating some matrix of pixels at a time. The convolution operation could be a dab item of the initial pixel values with weights characterized within the channel. The outcomes form past layer is summed up into one number that speaks about all the pixels the channel observed. Activation layer is a lattice that the convolution layer creates with lesser in estimate than the first picture. This framework is run through an enactment layer, which presents non-linearity to permit the arrange to prepare itself through backpropagation. The enactment work is ordinarily (Rectified Linear Unit)ReLu. Pooling layer applies the method of "pooling" to encourage down sampling and diminish the measure of the framework. A channel is passed over that comes out of the past layer and chooses one number out of each bunch of values (regularly the most extreme and this is typically called max pooling). This permits the CNN organization to prepare much quicker, focusing on the most critical information in each feature of the image. Fully connected layer is a traditional multilayer perception structure. Its input is a one-dimensional vector representing the output of the previous layerswhere there are a list of probabilities for different possible labels attached to the image (e.g. dog, cat or bird). The label that receives the highest probability is the classification decision.

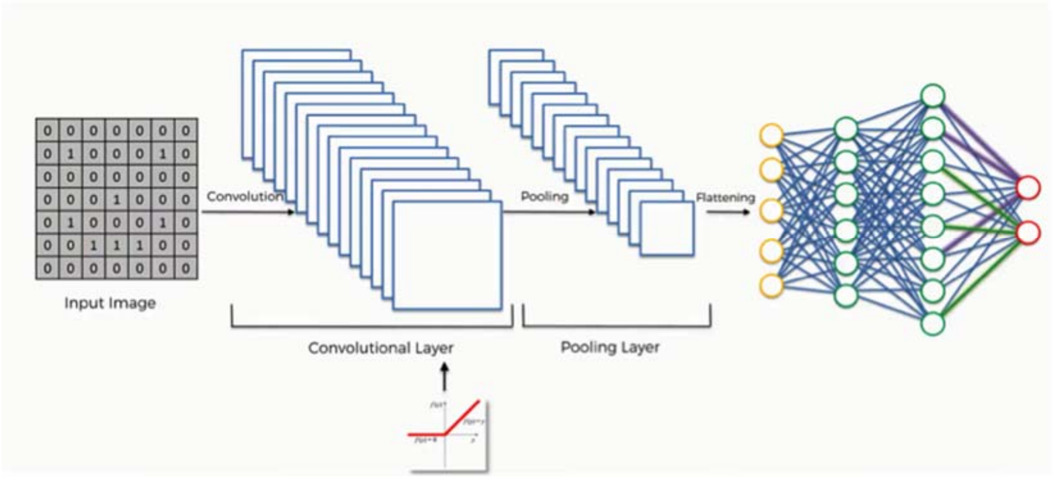


Fig 1: Architecture of a Convolution Neural Network.

The CNN is a state-of-the-art method for the image classification was first attempted to make a CNN model for the stanas dataset. However, the dataset was too small and the performance of the CNN was insufficient. Contrary to this, instead of manually extracting the features, we tried to obtain the features using transfer learning. Transfer learning involves the approach where the information gained is transferred to one or more source tasks and is used to enhance the learning of a similar target task.

While most machine learning algorithms are designed to address single tasks, a subject of ongoing interest in machine learning is to develop algorithms that promote transfer learning. It is assumed that fully-connected layers collect information relevant to solving a particular problem. For example, the fully connected layers of Alex Net will show the features that are essential for classifying an image into one of 1000 categories of artifacts. For a new problem, one can use the last layer features of a state-of-the-art CNN that was pre-trained on ImageNet and create a new model on these extracted features. In practice, it is recommended to either keep the pre-trained parameters fixed or tune them with a small learning rate in order to ensure that the system does not unlearn the previously acquired knowledge. Figure 2 shows a transfer learning[8] model with a pre-trained deep learning model as a feature extractor.

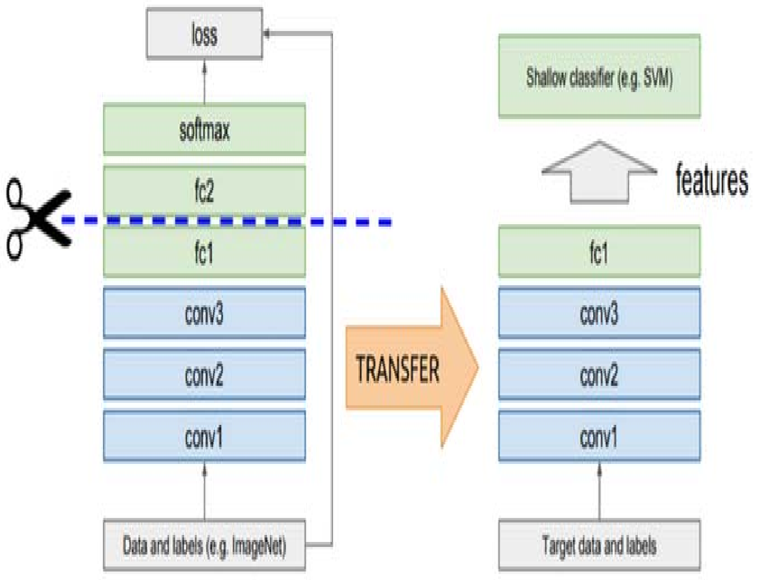


Fig 2: Transfer Learning with pre-trained Deep Learning Models as Feature Extractor

The mathematical definition of transfer learning[9] is given in terms of domain and task. The domain consists of D a feature space χ and a marginal probability distribution $P\ (X\ )$ where $X\ =\ \{x_1\ ,\ldots,x_n\} \in\ \chi$ is a specific domain, D = $\{\ \chi\ ,P\ (\ X\ )\}$ is a task that consists of two components: a label space Y and an objective predictive function $f\ (\cdot)$(denoted by $T\ =\ \{Y\ ,f\ (\cdot)\}$. $f\ (\cdot)$ is acquired from the training data consisting of pairs$\{\ x_i\ ,y_i\ \}$, where $x_i \in\ X$ and $y_i\ \in\ Y$. The function $f\ (\cdot)$can be used to predict the corresponding label while $f\ (\ x\ )$uses a new instance, x. Given a source domain D_S and learning task T_S, a target domain D_T and learning task T_T, transfer learning aims to help improve the learning of the target predictive function $f_T\ (\cdot)$in D_T using the knowledge in D_S and T_S where $D_S \neq\ D_T$ or $T_S \neq\ T_T$.

4. Methodology

4.1. Building Deep Stana Classification

The problem addressed in this paper is modular as it can be divided into two non-overlapping modules. One module identified whether a particular stana is present in an image and the other is a video processing module that processed the video and generated an annotation schema. In the preliminary phase of work, a machine learning classifier was built for classification of dance images into six different stanas like Vaishnava, Samapada, Vaishakha, Mandala, Alidha, and Prathyalida as shown in figure 3

+

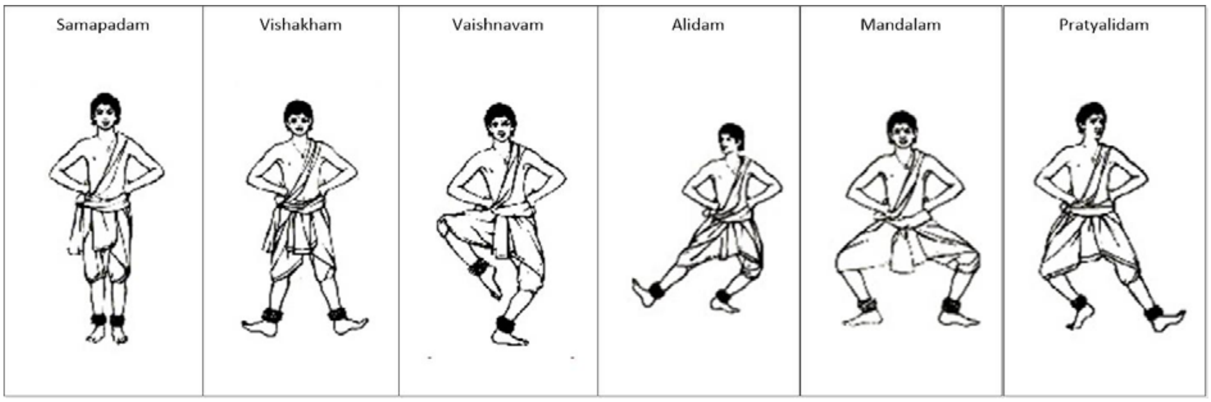
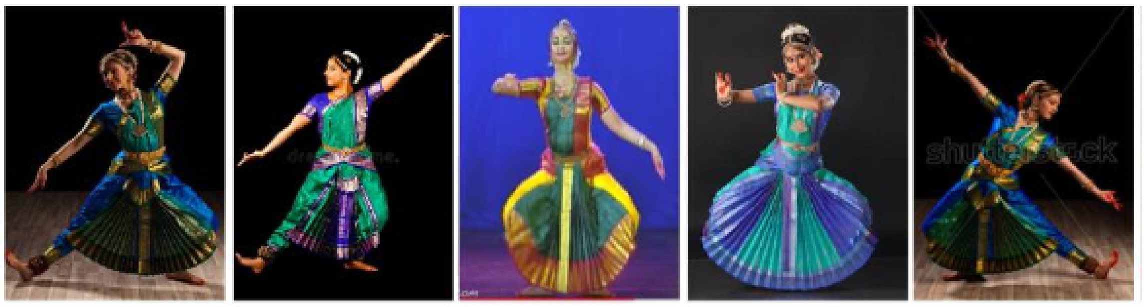


Fig 3: Architecture of Deep Stana Classification



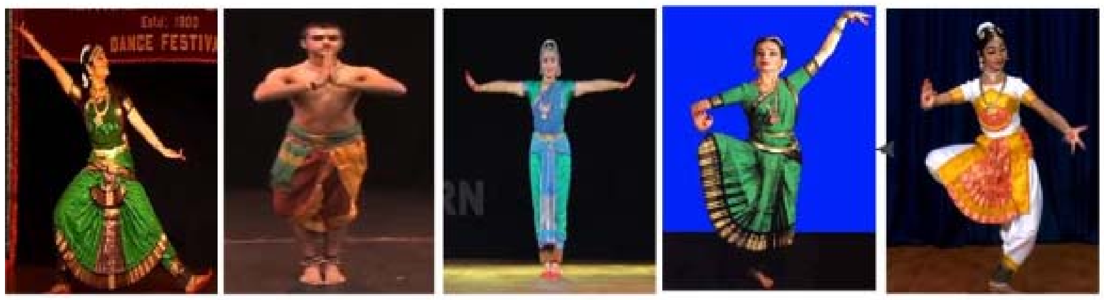


Fig 4: Different stanas

From each image eight geometric values and local shape descriptors like hu-moment were extracted as features then different classification algorithms were applied. Among them, the Naïve Bayes classifier was chosen as the best algorithm for stana image classification as it gives more classification accuracy. For the above work, animage dataset of nearly 500 images with equal distribution to each class was created and using a novel feature extraction algorithm, features were explicitly clamped out and the classifier was built. The overall architecture of the Deep Stana Classifier is given in Figure 4. The dataset used in this work has approximately 600 pictures that consists of seven classes, six stanas classes, and one class with other images frames that can occur in a dance sequence. As transfer learning is used for feature extraction, the preprocessing phase does not need any hardcore image processing operations. All the images are resized into 224 X 224, which is found to be an optimal through trial and error. When implementing the Classifier, a pre-trained transfer learning model, ImageNet, along with VGG19[10] image dataset, was used. The ImageNet model is shown in Figure 5[11].

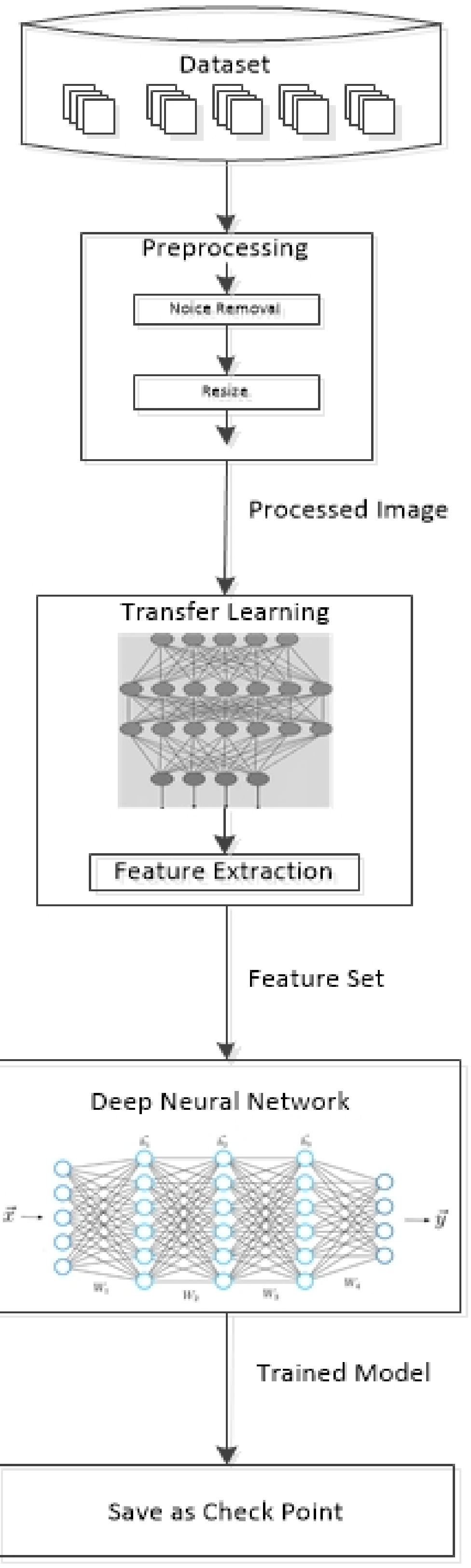


Fig 5: VGG19 ImageNet

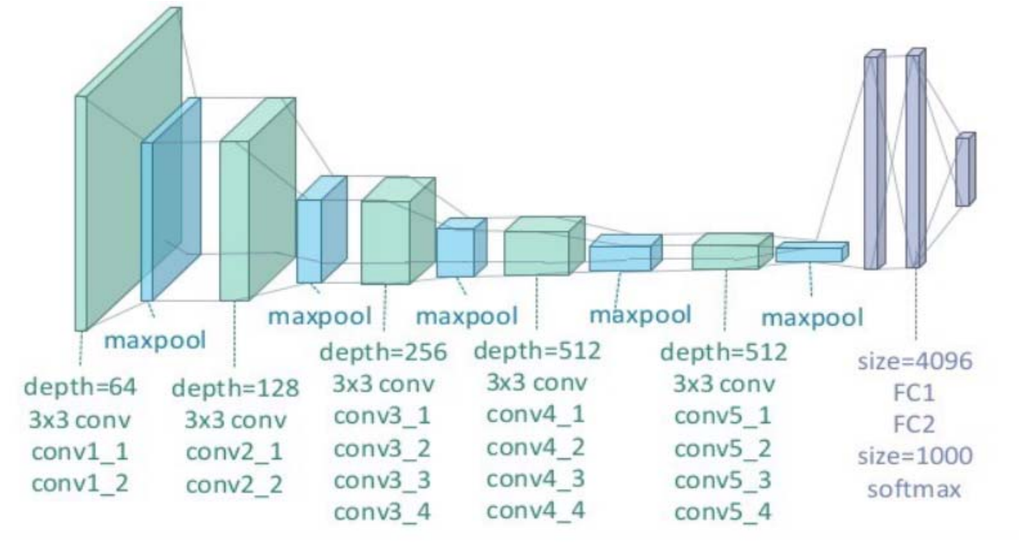


Table 1 : DNN model summary

Layer (type)	Output shape	Param
Dense_1 (Dense)	(None, 10000)	250890000
Dense_2 (Dense)	(None, 5000)	500050000
Dense_3 (Dense)	(None, 2500)	12502500
Dense_4 (Dense)	(None, 1000)	2501000
Dense_5 (Dense)	(None, 500)	125250
Dense_6 (Dense)	(None, 250)	30120
Dense_7 (Dense)	(None, 30)	3630
Dense_8 (Dense)	(None, 30)	930
Dense_9 (Dense)	(None, 7)	217
Total params: 316,559,147 Trainable params: 316,559,147 Non-trainable params: 0		

A deep neural network has a pyramidal structure. It consists of an input layer with 25088 input nodes, eight hidden layers all are fully connected, and an output layer with seven nodes as in figure 7. The tensorflow model summary is shown in Table 1. Deep learning libraries like Keras[12] and Tensorflow provide smart APIs for integrating transfer learning to the proposed model. From an image of dimension 224 X 224, a total of 25088 features were extracted and were input into the classifier for building a classification model. During training, the entire dataset is divided into training and testing sets. The training set is initially fed to the model, which is designed to iterate 300 times specified by epochs and take 64 datapoints as a batch for a single iteration. Along with the training, validation is also completed at each iteration. After the required number of iterations is over, the test data is given to the classifier to evaluate the model accuracy. The trained model is stored as checkpoints to determine whether the accuracy meets the requirement. In the next phase of the work for implementing the stanas video annotator, the above-trained DNN model is used.

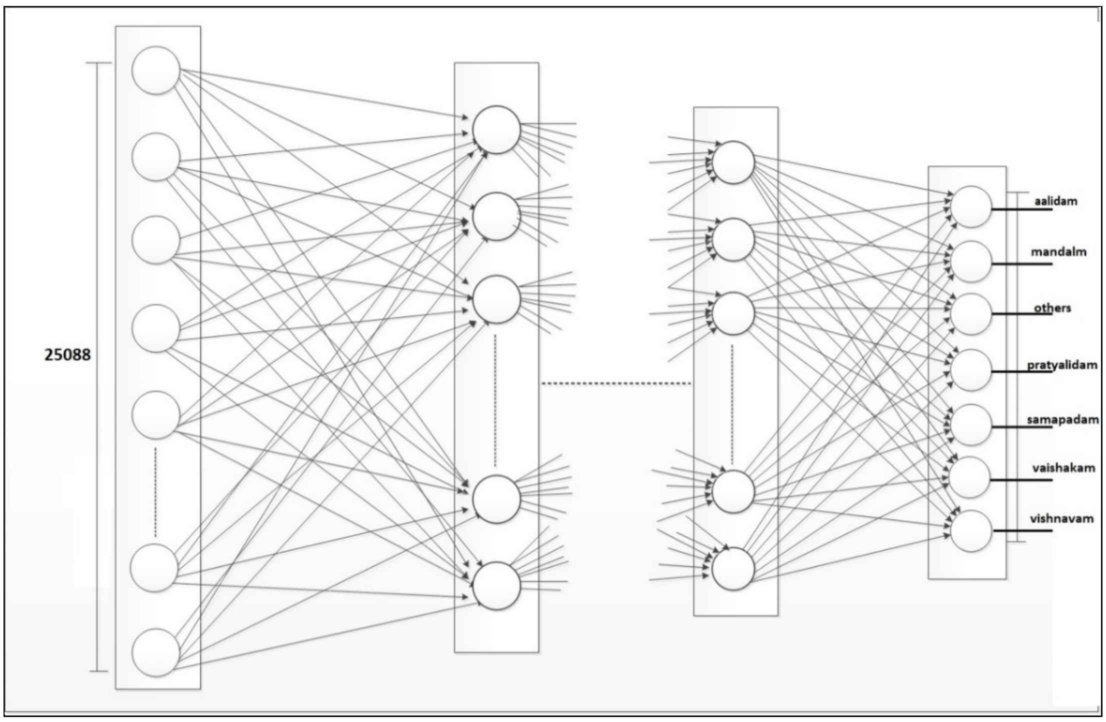


Fig 7: Structure of network

4.2. Stana Identification and Video Annotation

The main objective of this phase is to take a video as input and process it for identifying the stanas in it. A video is a sequence of images encoded and compressed in formats like MPEG, FLV, AVI etc. However, in this work, the compressed sequence of images cannot be used for the annotation process, and instead, each frame of the video is taken as a complete uncompressed image. From the image sequence, the frame-fetch module draws a frame for processing, which is dissimilar to the previously processed image. This is done by finding the relative difference through frame-to-frame subtraction. If the difference is relatively significant, then the image frame will be taken for further processing. The frame picked up by a frame-fetch module is given to a pre-trained classifier. The classifier, in turn, predicts the input image into one of the seven stana classes. If the predicted class belongs to one of the stanas, then the frame along with time-stamp and frame-number will be indexed. If the frame does not contain any identifiable stana class and the classifier predicts it into ‘others’ class, then the frame is neglected and the next frame is fetched from the video. By extending this process, the entire video can be tagged and can create an index according to stana.

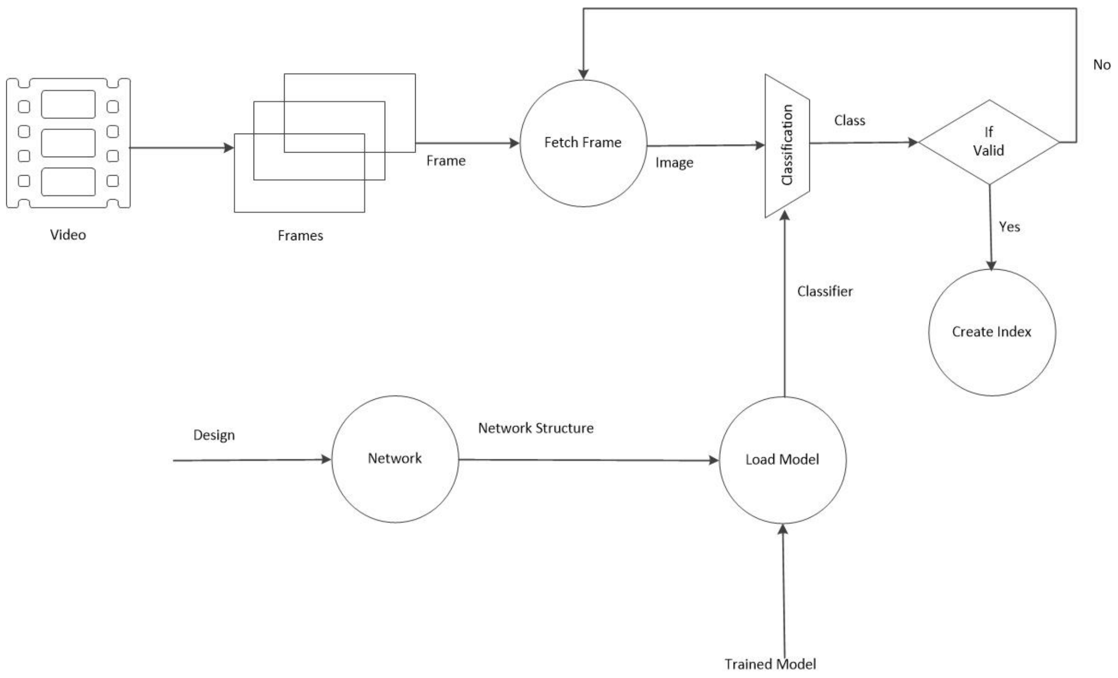


Fig 8: Annotation Process

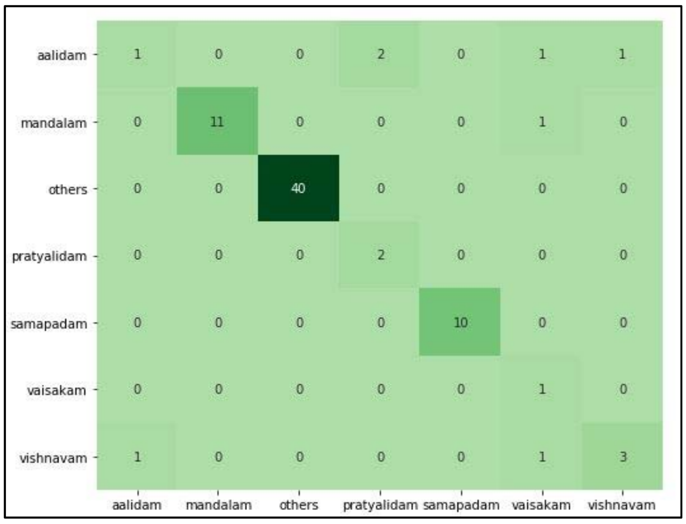
5. Experimental Setup and Result Analysis

This section includes the practical implementation and the analysis of the theoretical aspects discussed earlier. The entire work is implemented as a deep learning application. The dataset consists of nearly 600 images, but not enough for a from-the-scratch implementation of the classifier. So, the benefits from the transfer learning technique are utilized. For implementation, the language used is very important and Python, along with packages like OpenCV[13], Keras, Tensorflow, Pandas, etc. are used.

For the analysis of this work, the priority is to analyze the working of the classifier model. While dealing with classification, different metrics like accuracy, precision, recall, and f1-score are to be calculated with the help of a confusion matrix. These metrics are shown in Figure 8 and Figure 9 with the help of a confusion matrix and classification report. It is clear from the classification report that the proposed methodology attained a remarkable classification accuracy of 91% compared to other classification methods. The index created by the annotation process of three sample videos is shown in Figure 1. The index structure is stored as a JSON object for quick processing and access.

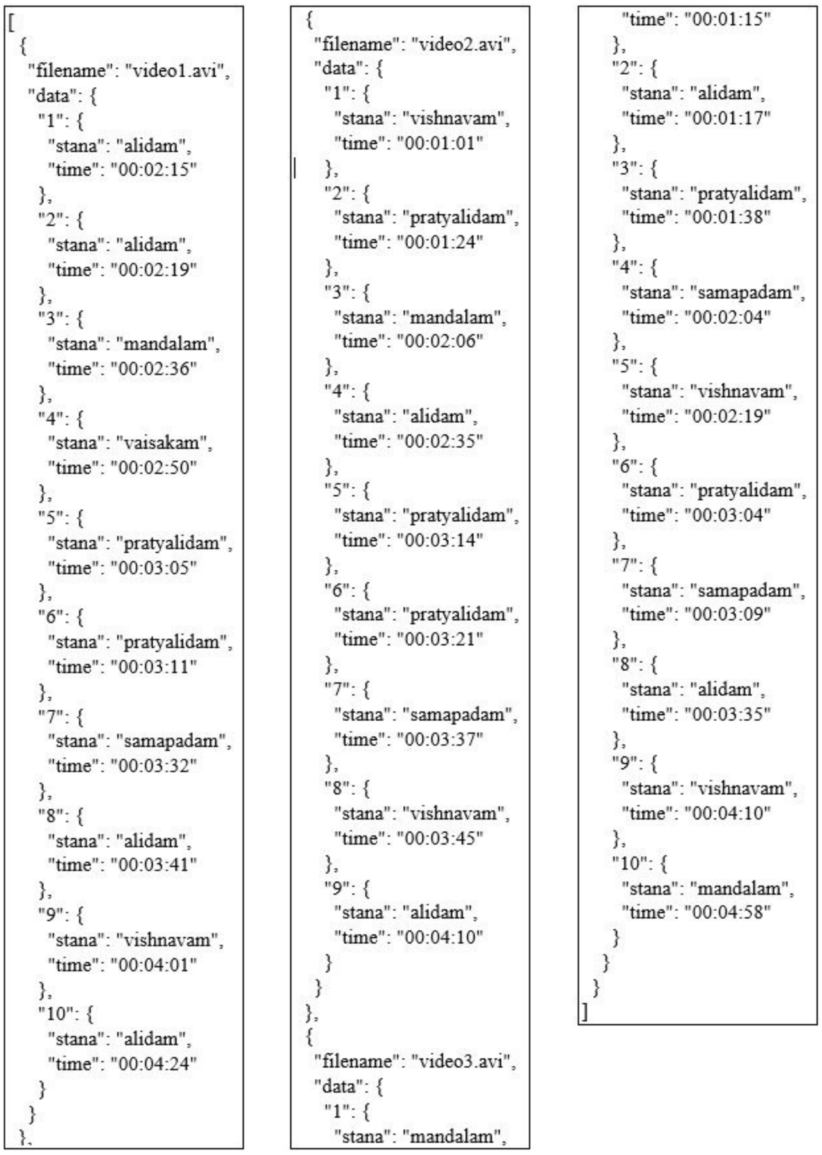
Table 2 Classification Report

	precision	**recall**	**f1-score**	**support**
0	0.5	0.20	0.29	5
1	1.00	.92	.96	12
2	1.00	1.00	1.00	40
3	0.50	1.00	0.67	2
4	1.00	1.00	1.00	10
5	0.25	1.00	0.40	1
6	0.75	0.60	0.67	5
Accuracy			0.91	75
macro avg	0.71	0.82	0.71	75
weighted avg	0.93	0.91	0.91	75



	aalidam	mandalam	others	pratyalidam	samapadam	vaisakam	vishnavam
aalidam	1	0	0	2	0	1	1
mandalam	0	11	0	0	0	1	0
others	0	0	40	0	0	0	0
pratyalidam	0	0	0	2	0	0	0
samapadam	0	0	0	0	10	0	0
vaisakam	0	0	0	0	0	1	0
vishnavam	1	0	0	0	0	1	3

Fig 10: Confusion Matrix



```
[
 {
  "filename": "video1.avi",
  "data": {
   "1": {
    "stana": "alidam",
    "time": "00:02:15"
   },
   "2": {
    "stana": "alidam",
    "time": "00:02:19"
   },
   "3": {
    "stana": "mandalam",
    "time": "00:02:36"
   },
   "4": {
    "stana": "vaisakam",
    "time": "00:02:50"
   },
   "5": {
    "stana": "pratyalidam",
    "time": "00:03:05"
   },
   "6": {
    "stana": "pratyalidam",
    "time": "00:03:11"
   },
   "7": {
    "stana": "samapadam",
    "time": "00:03:32"
   },
   "8": {
    "stana": "alidam",
    "time": "00:03:41"
   },
   "9": {
    "stana": "vishnavam",
    "time": "00:04:01"
   },
   "10": {
    "stana": "alidam",
    "time": "00:04:24"
   }
  }
 },
 {
  "filename": "video2.avi",
  "data": {
   "1": {
    "stana": "vishnavam",
    "time": "00:01:01"
   },
   "2": {
    "stana": "pratyalidam",
    "time": "00:01:24"
   },
   "3": {
    "stana": "mandalam",
    "time": "00:02:06"
   },
   "4": {
    "stana": "alidam",
    "time": "00:02:35"
   },
   "5": {
    "stana": "pratyalidam",
    "time": "00:03:14"
   },
   "6": {
    "stana": "pratyalidam",
    "time": "00:03:21"
   },
   "7": {
    "stana": "samapadam",
    "time": "00:03:37"
   },
   "8": {
    "stana": "vishnavam",
    "time": "00:03:45"
   },
   "9": {
    "stana": "alidam",
    "time": "00:04:10"
   }
  }
 },
 {
  "filename": "video3.avi",
  "data": {
   "1": {
    "stana": "mandalam",
    "time": "00:01:15"
   },
   "2": {
    "stana": "alidam",
    "time": "00:01:17"
   },
   "3": {
    "stana": "pratyalidam",
    "time": "00:01:38"
   },
   "4": {
    "stana": "samapadam",
    "time": "00:02:04"
   },
   "5": {
    "stana": "vishnavam",
    "time": "00:02:19"
   },
   "6": {
    "stana": "pratyalidam",
    "time": "00:03:04"
   },
   "7": {
    "stana": "samapadam",
    "time": "00:03:09"
   },
   "8": {
    "stana": "alidam",
    "time": "00:03:35"
   },
   "9": {
    "stana": "vishnavam",
    "time": "00:04:10"
   },
   "10": {
    "stana": "mandalam",
    "time": "00:04:58"
   }
  }
 }
]
```

Fig 11. Index in JSON format

6. Conclusion and Future Work

The need for digitization and archiving art forms has only become relevant and resources related to it like dance videos contain a much complex form of data. The archive is useful only when there is provision for the efficient retrieval. The aim of this paper was to annotate and create an index for SICD videos based on stanas or foot postures, which is an essential element in classical dance. The first part of the paper elaborated on the development of a Deep Stana Classifier. The features from the images are extracted using transfer learning, and a deep neural network is used to identify them. The Deep Stana Classifier trained model is then used to classify stanas from the frames of a video. The Deep Stana Classifier's accuracy is quite promising because it identified 91% of the foot work from the video. The second part of the work was to develop an automatic annotation for the video through an annotation module where information is maintained in a JSON format. The primary focus of the paper was to understand foot postures or stanas, but there are many aspects in classical dance where a model of the Deep Classifier can be applied, such as in constructs like hasta bedas, karanas, etc. Most of these constructs can be analyzed in a spatio-temporal manner using advanced deep learning concepts like recurrent neural networks (RNN) and long-short term memory (LSTM).

7. References

[1] H. Chaudhry, K. Tabia, S. A. Rahim, and S. Benferhat, “Automatic annotation of traditional dance data using motion features,” 2nd Jt. Int. Conf. Digit. Arts, Media Technol. 2017 Digit. Econ. Sustain. Growth, ICDAMT 2017, pp. 254–258, 2017.
[2] B. Ramadoss and K. Rajkumar, “Generic modeling and annotation of the dance video semantics,” in Proceedings - Sixth IEEE International Conference on Computer and Information Technology, CIT 2006, 2006.
[3] B. Ramadoss and K. Rajkumar, “Semi-automated annotation and retrieval of dance media objects,” Cybern. Syst., vol. 38, no. 4, pp. 349–379, 2007.
[4] B. Ramadoss and K. Rajkumar, “Modeling the Dance Video annotations,” in 2006 1st International Conference on Digital Information Management, ICDIM, 2006, pp. 145–150.
[5] L. E. I. Chen and M. T. Özsu, “Modeling video data for content based queries: extending the disima image data model,” 1997.
[6] A. Ekin, A. M. Tekalp, and R. Mehrotra, “Integrated Semantic – Syntactic Video Modeling for Search and Browsing,” vol. 6, no. 6, pp. 839–851, 2004.
[7] S. Indolia, A. K. Goswami, S. P. Mishra, and P. Asopa, “Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach,” Procedia Comput. Sci., vol. 132, pp. 679–688, 2018.
[8] L. Pratt and B. Jennings, “A Survey of Transfer Between Connectionist Networks,” Conn. Sci., vol. 8, no. 2, pp. 163–184, 1996.
[9] Y. P. Lin and T. P. Jung, “Improving EEG-based emotion classification using conditional transfer learning,” Front. Hum. Neurosci., vol. 11, no. June, pp. 1–11, 2017.
[10] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc., pp. 1–14, 2015.
[11] Y. Zheng, C. Yang, and A. Merkulov, “Breast cancer screening using convolutional neural network and follow-up digital mammography,” 2018, no. September, p. 4.
[12] F. and others Chollet, “Keras,” 2015. [Online]. Available: https://keras.io.
[13] G. Bradski, “opencv_library,” Dr. Dobb’s J. Softw. Tools, 2000.