Indian Dance Forms Classification

Introduction

Indian classical dance, or ICD, is a reflection of the country's rich cultural legacy. Each dance originates from a distinct state within the nation. Nine distinct dance forms are recognized by https://www.indiaculture.gov.in/dance [Ministry of Culture, Government of India]. These dance forms include Bharatanatyam, Kathak, Kathhakali, Kuchipudi, Manipuri, Odissi, Sattriya and Chhau. Previous work in the paper "Classification of Indian Dance Forms using Pre-Trained Model-VGG" [1] was done for the eight dance forms excluding Chhau and employed transfer learning to address this multiclass classification in this project. Images of these eight dance forms that are known in India were classified using pre-trained models like VGG16 and VGG19. The ICD dataset from Kaggle, which included several photos in each of the eight classes was used.

Literature Survey

In [2] the authors used a Computer Vision techniques and SVM to classifiy Indian Classical Dance (ICD). They used a sparse representation based dictionary learning technique. First, represent each frame of a dance video by a pose descriptor based on histogram of oriented optical flow (HOOF), in a hierarchical manner. The pose basis is learned using an on-line dictionary learning technique. Finally each video is represented sparsely as a dance descriptor by pooling pose descriptor of all the frames. In their work, dance videos are classified using support vector machine (SVM) with intersection kernel. An accuracy of 86.67% was achieved while tested their algorithm on their own ICD dataset created from the videos collected from YouTube.

In [3] the authors used Deep Learning Convolution Neural Network (CNN) to classify five dance classes namely Bharatanatyam, Odissi, Kathak, Kathakali, Yakshagana, the images of which are collected from the internet using Google Crawler.

Transfer Learning

Transfer learning is a machine learning technique where knowledge learned from a task is re-used to boost performance on a related task. It is a popular approach in deep learning as it enables the training of deep neural networks with less data compared to having to create a model from scratch. The process involves unfreezing some layers of the model and then retraining it at a low-learning rate to handle a new dataset. Transfer learning is not a distinct type of machine learning algorithm, but rather a technique or method used whilst training models. By harnessing the ability to reuse existing models and their knowledge of new problems, transfer learning has opened doors to training deep neural networks even with limited data.

VGG 16

VGG-16, or the Visual Geometry Group 16-layer model, is a deep convolutional neural network architecture that was proposed by the Visual Geometry Group at the University of Oxford. It was introduced in the paper titled "Very Deep Convolutional Networks for Large-Scale

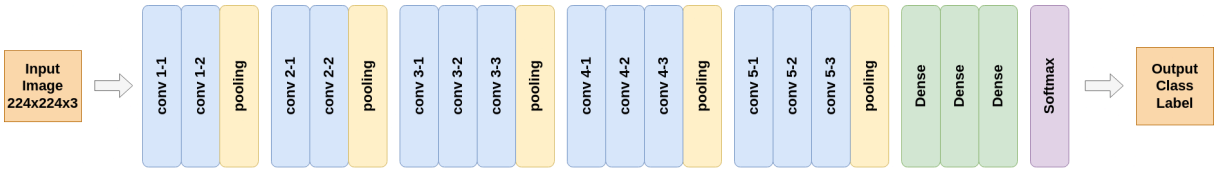

Figure 1: VGG-16 architecture Map

Image Recognition" by K. Simonyan and A. Zisserman in 2014 [4]. VGG-16 is a part of the VGG family of models, which also includes VGG-19, VGG-M, and others.

Architecture

The VGG-16 architecture is characterized by its simplicity and uniformity. It consists of 16 weight layers, including 13 convolutional layers and 3 fully connected layers. The convolutional layers are followed by max-pooling layers, and the fully connected layers are followed by a softmax activation function for classification.

The general architecture is as follows:

- **Input Layer** (224x224x3): Accepts an RGB image with a resolution of 224x224 pixels.
- **Convolutional Layers** (Conv):
  - The first two layers have 64 filters with a 3x3 kernel and a stride of 1.
  - The next two layers have 128 filters with a 3x3 kernel and a stride of 1.
  - The next three layers have 256 filters with a 3x3 kernel and a stride of 1.
  - The final three layers have 512 filters with a 3x3 kernel and a stride of 1.
- **Max-Pooling Layers** (MaxPool):
  - After every two convolutional layers, max-pooling with a 2x2 window and a stride of 2 is applied.
  - The goal of the layer of pooling is to sample based on the input dimension. This decreases by this activation the number of parameters.
- **Fully Connected Layers** (FC):
  - There are three fully connected layers, each with 4096 neurons.
  - The last layer has 1000 neurons, corresponding to the 1000 ImageNet classes.
- **Softmax Layer**
  - The final layer uses the softmax activation function for classification.

Table 1 shows the output shape and number of parameters for each layer used in VGG16 model.

Key Features:

- **Uniform Filter Size**

  VGG-16 uses a small 3x3 filter size throughout the convolutional layers, providing a uniform receptive field.

- **Deep Network**

  With 16 layers, VGG-16 is considered a deep neural network. The depth contributes to the model's ability to learn hierarchical features.

- **Ease of Interpretability**

  The uniform structure and simplicity of VGG-16 make it easy to interpret and understand compared to more complex architectures.

Dataset

We obtained from Kaggle the ICD dataset [5], which included 599 images from 8 distinct classes: Manipuri, Bharatanatyam, Odissi, Kathakali, Kathak, Sattriya, Kuchipudi, and Mohiniyattam.

Purulia Chhau, the ninth class of dance, has also been added. Since the majority of the pictures of chhau that are available online are of the purulia form, we only added this subdance form. With the addition of Purulia Chhau a total of 653 images are present in the dataset.

Training

VGG-16 was pretrained on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) dataset, a large dataset containing millions of labeled images across thousands of classes.

First, the data will be divided into training, testing, and validation samples. The VGG16 model will then be trained using our training set of data. A portion of it will be utilized at the validation stage. Next, using testing samples, we will test the model.

Performance Measurements

We further measured the performance of our model using a confusion matrix. The main idea behind this tabular performance measurement is to understand how good our model is when dealing with false-positives and false-negatives. Comparison between the actual and predicted classes helped us to visualize the accuracy of our model. In a confusion matrix, the true positive indicates the values which are predicted correctly predicted as positive whereas the false positive refers to the negative values which are incorrectly predicted as positive values. Similarly, false negative tells us the number of positive values which are predicted as negative by our model and lastly, the true negative refers to the negative values which are correctly predicted as negative.

Accuracy is considered as one of the most important factors to measure the performance. It is the ratio of correctly predicted observation to the total number of observations. Mathematically,

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}}$$

Next factor is precision. It refers to the ratio of correctly predicted positive values to the total predicted positive values. Mathematically,

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Recall indicates to the ratio of correctly predicted positive observations to the all observations in actual class. Mathemat- ically,

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 224, 224, 3)]	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
global_average_pooling2d (GlobalAveragePooling2D)	(None, 512)	0
fc-1 (Dense)	(None, 4096)	2101248
dropout (Dropout)	(None, 4096)	0
fc-2 (Dense)	(None, 4096)	16781312
dropout_1 (Dropout)	(None, 4096)	0
output_layer (Dense)	(None, 9)	36873
Total params		33634121 (128.30 MB)
Trainable params		18919433 (72.17 MB)
Non-trainable params		14714688 (56.13 MB)

Table 1: Summary of VGG-16.

Figure 2: Dance forms images from the datasets.

	Class	Images
0	bharatanatyam	73
1	kathak	76
2	kathakali	74
3	kuchipudi	76
4	manipuri	84
5	mohiniyattam	70
6	odissi	71
7	sattriya	75
8	purulia chhau	53

Table 2: Number of images in each class

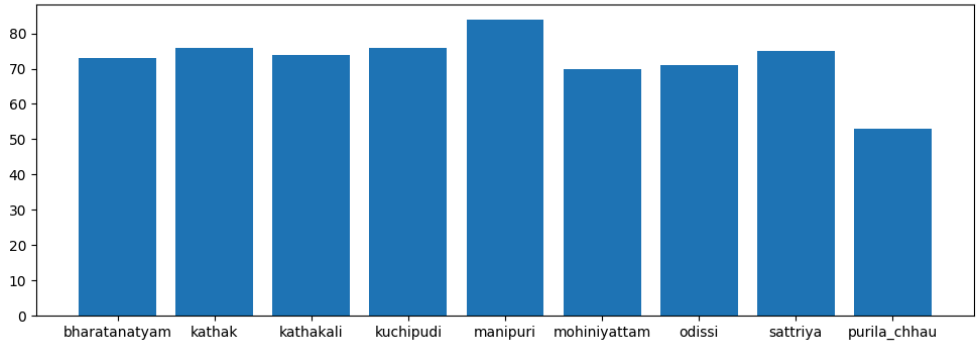

Figure 3: Number of images in each class

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Lastly, F1 Score is the weighted average of Precision and Recall. Mathematically,

$$\text{F1 Score} = 2 * \frac{\text{Recall} * \text{Precision}}{\text{Recall} + \text{Precision}}$$

Table 4 shows a sample confusion matrix for binary classification. Table 5 shows the parameters of evaluation and Table 6 shows the confusion matrix for the VGG16 model trained and tested on our dataset.

The number of epochs is a hyperparameter. We have implemented our models using 30 epochs for VGG16. The number of times the learning algorithm will work through the entire training data is referred as epoch. We have plotted the train_loss vs val_loss and train_acc vs val_acc for VGG 16 model.

Figure 4 shows the training accuracy vs validation accuracy for different epochs. Figure 5 shows the training loss vs validation loss for different epochs.

Epochs	Training Accuracy	Val. Accuracy	Training Loss	Val. Loss
1	0.11	0.1	14.25	15.04
2	0.11	0.1	14.27	15.04
3	0.11	0.1	14.27	15.04
4	0.11	0.1	14.27	15.04
5	0.11	0.1	14.27	15.04
6	0.11	0.1	14.27	15.04
7	0.11	0.1	14.27	15.04
8	0.11	0.1	14.27	15.04
9	0.11	0.1	14.27	15.04
10	0.11	0.1	14.27	15.04
11	0.11	0.1	14.27	15.04
12	0.11	0.1	14.27	15.04
13	0.11	0.1	14.27	15.04
14	0.11	0.1	14.27	15.04
15	0.11	0.1	14.27	15.04
16	0.11	0.1	14.27	15.04
17	0.11	0.1	14.27	15.04
18	0.11	0.1	14.26	15.04
19	0.11	0.1	14.27	15.04
20	0.11	0.1	14.27	15.04
21	0.11	0.1	14.27	15.04
22	0.11	0.1	14.27	15.04
23	0.11	0.1	14.27	15.04
24	0.11	0.1	14.27	15.04
25	0.11	0.1	14.27	15.04
26	0.11	0.1	14.27	15.04
27	0.11	0.1	14.27	15.04
28	0.11	0.1	14.27	15.04
29	0.11	0.1	14.27	15.04
30	0.11	0.1	14.27	15.04

Table 3: Training VGG16 model.

True Positives (TP)	**False Negatives (FN)**
False Positives (FP)	**True Negatives (TN)**

Table 4: Confusion Matrix

Class	Precision	Recall	F1 score	Support
0	0.87	0.81	0.84	16
1	0.79	0.65	0.71	17
2	1.00	0.89	0.94	9
3	0.72	0.87	0.79	15
4	0.83	1.00	0.90	19
5	1.00	1.00	1.00	10
6	0.76	0.93	0.84	14
7	0.95	0.82	0.88	9
8	1.00	0.78	0.88	9
Accuracy			0.85	131
Macro avg	0.88	0.86	0.86	131
Weighted avg	0.86	0.85	0.85	131

Table 5: Classification Report

	0	1	2	3	4	5	6	7	8
0	13	1	0	2	0	0	0	0	0
1	1	11	0	1	1	0	2	1	0
2	0	0	8	0	1	0	0	0	0
3	0	1	0	13	0	0	1	0	0
4	0	0	0	0	19	0	0	0	0
5	0	0	0	0	0	10	0	0	0
6	1	0	0	0	0	0	13	0	0
7	0	1	0	1	1	0	1	18	0
8	0	0	0	1	1	0	0	0	7

Table 6: Confusion Matrix

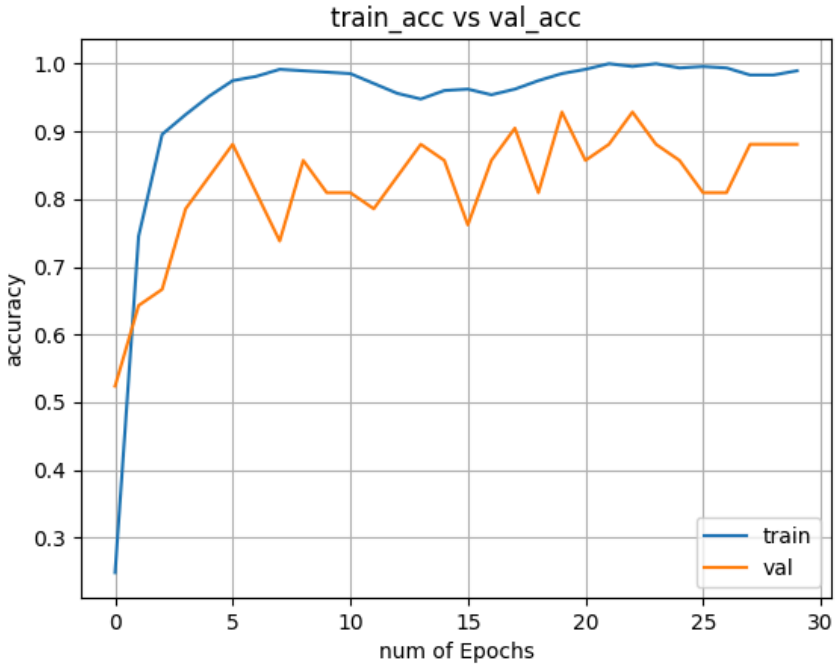

Figure 4: Training accuracy vs validation accuracy for VGG-16.

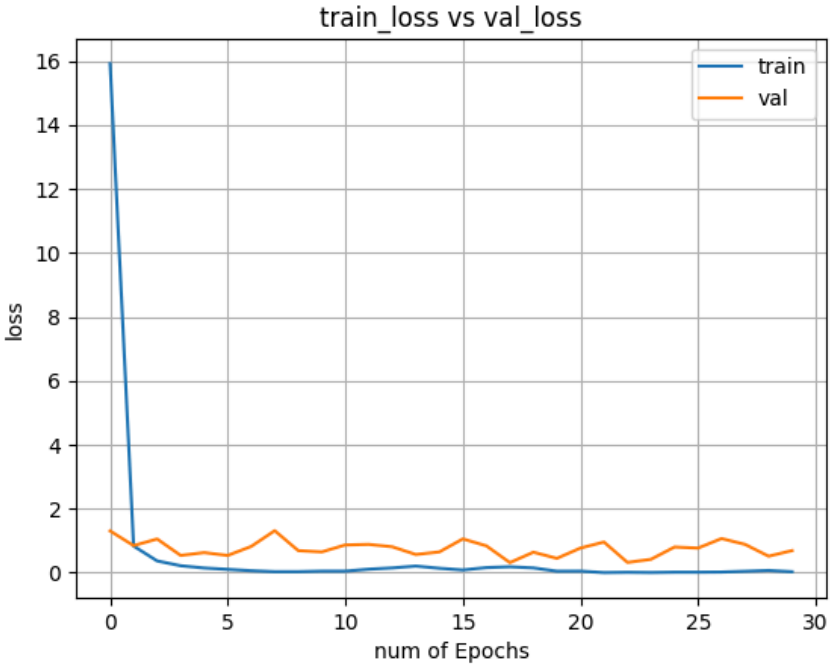

Figure 5: Training loss vs validation loss for VGG-16.

References

[1] S. Biswas, A. Ghildiyal, and S. Sharma, "Classification of Indian Dance Forms using Pre-Trained Model-VGG," in *2021 Sixth International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, 2021, pp. 278–282. doi: 10.1109/WiSPNET51692.2021.9419426.

[2] S. Samanta, P. Purkait, and B. Chanda, "Indian Classical Dance classification by learning dance pose bases," in *2012 IEEE Workshop on the Applications of Computer Vision (WACV)*, 2012, pp. 265–270. doi: 10.1109/WACV.2012.6163050.

[3] A. D. Naik and M. Supriya, "Classification of Indian Classical Dance Images using Convolution Neural Network," in *2020 International Conference on Communication and Signal Processing (ICCSP)*, 2020, pp. 1245–1249. doi: 10.1109/ICCSP48568.2020.9182365.

[4] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv preprint arXiv:1409.1556*, 2015, doi: 10.48550/arXiv.1409.1556.

[5] Aditya, "Indian Dance form Classification retrieved from https://www.kaggle.com/aditya48/indian-dance-form-classification." 2020.