Yoga Pose Classification Using Deep Learning and Web Application Deployment

Anonymous submission

Paper ID

Abstract

This project presents a novel approach for classifying yoga poses using a deep learning model integrated with a user-friendly web application. We address the challenge of accurately identifying yoga poses, which are complex due to subtle variations in human postures. Our model achieves significant accuracy, demonstrating effectiveness in real-world scenarios. The integration into a web app makes this accessible for end-users, facilitating practical application. Experimental results show a high classification accuracy, indicating the model's robustness in handling diverse yoga pose datasets.

1. Introduction

Yoga poses vary subtly in orientation and structure, posing a challenging classification problem for machine learning models. Accurately classifying these poses has applications in health monitoring, fitness coaching, and personal training. Previous works have largely focused on general human pose estimation, but our approach is tailored specifically for the nuances of yoga postures. We propose a deep learning-based classifier that leverages convolutional neural networks to identify distinct yoga poses accurately. The integration of this classifier into a web application allows for easy accessibility and real-time feedback.

2. Proposed Method

Our approach consists of three main components: data preprocessing, model architecture, and deployment strategy.

2.1. Data Preprocessing

We utilized a dataset of yoga pose images, ensuring diversity in pose variations and lighting conditions. The preprocessing pipeline includes image resizing, normalization, and data augmentation (rotation, zoom, and horizontal flip) to enhance the model's generalizability.

2.2. Angle Calculation

Given three points A, B, and C, the angle θ formed at point B is determined using vector geometry. The following steps outline the process:

1. **Vector Representation**: Define the vectors:

$$\overrightarrow{BA} = \mathbf{a} - \mathbf{b}, \quad \overrightarrow{BC} = \mathbf{c} - \mathbf{b} \tag{1}$$

   where $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$ are the coordinates of points A, B, and C, respectively.

2. **Cosine of the Angle**: The cosine of the angle θ between the vectors $\overrightarrow{BA}$ and $\overrightarrow{BC}$ is given by the formula:

$$\cos(\theta) = \frac{\overrightarrow{BA} \cdot \overrightarrow{BC}}{\|\overrightarrow{BA}\| \|\overrightarrow{BC}\|} \tag{2}$$

   where $\overrightarrow{BA} \cdot \overrightarrow{BC}$ is the dot product of the vectors and $\|\overrightarrow{BA}\|$ and $\|\overrightarrow{BC}\|$ are the magnitudes (norms) of the vectors.

3. **Angle in Radians**: The angle θ in radians is calculated using the inverse cosine (arccos) function:

$$\theta = \arccos\left(\cos(\theta)\right) \tag{3}$$

4. **Conversion to Degrees**: Finally, the angle is converted from radians to degrees:

$$\theta_{\text{deg}} = \frac{\theta_{\text{rad}} \times 180}{\pi} \tag{4}$$

2.3. Body Posture Analysis Using MediaPipe Pose Estimation

MediaPipe's pose estimation capabilities for analyzing body posture is discussed further. The approach involves initializing the model, preparing data, processing images, calculating joint angles and distances, and compiling the results.

2.3.1 Pose Model Initialization

The MediaPipe pose detection model is utilized to accurately identify body landmarks in images. The model is configured to operate in static image mode, with a detection confidence threshold set to improve landmark accuracy in static photos. This initialization ensures each image is treated independently, reducing error rates and enhancing detection robustness.

2.4. Dataset - Yoga82

The dataset-Yoga 82 is used in our work . Yoga-82, a dataset designed for large-scale yoga pose recognition with 82 classes, consists of complex poses where fine annotations is not be possible. Hierarchical labels are provided for yoga poses based on the body configuration of the pose. The dataset contains a three-level hierarchy including body positions, variations in body positions, and the actual pose names.

The dataset has a varying number of images in each class, ranging from 64 (min.) to 1133 (max.), with an average of 347 images per class. Some images are sourced from a specific yoga website and contain yoga poses with clean backgrounds. However, there are also images with diverse backgrounds, such as forests, beaches, and indoor settings. Some images feature only silhouettes, sketches, or drawings of yoga poses. These are included in the dataset as yoga poses are primarily focused on the overall structure of the body rather than the texture of clothes or skin. For ease of understanding and readability, only English names are used for the yoga poses in this document. However, Sanskrit names are also available in the dataset for reference.

2.4.1 Data Preparation

The Yoga-82 dataset is first extacted by downloads images corresponding to various yoga poses using URLs listed in text files located within a specified folder. Each text file represents a unique yoga pose and contains lines with a filename and a URL separated by a tab. For each file, we have generated a sub-folder named after the yoga pose in a main dataset directory. Each URL is read ,accessibility is verified , and the images are downloaded if available. To prevent duplicate images,a unique hash is used for each image file and only saves it if it does not already exist within the folder . The entire dataset is not used for our work we have applied a condition and only used a part of dataset for our work. We have skipped the images whose URLs were not accessible. Data structures are organized to facilitate efficient processing and storage: - Each pose is named and stored in a dedicated list, aiding in clear categorization. - Computed joint angles and distances are systematically stored for further analysis. - Labels corresponding to each body posture

Figure 1. Some poses in Yoga-82 dataset

are included for accurate classification in later stages.

The dataset path is also structured to categorize images by distinct poses, streamlining access and processing.

2.4.2 Image Processing Pipeline

Each image in the dataset undergoes a series of transformations to prepare it for pose analysis: - Images are converted to the RGB color space to match the input requirements of the MediaPipe model. - The pose model detects and records body landmarks, outputting precise 3D coordinates for key anatomical points, assuming successful landmark identification.

2.4.3 Joint Angle and Distance Calculation

Using the detected landmarks, **joint angles and distances** are calculated to capture the posture characteristics for each pose. Key calculations include:

- **Joint Angle Measurement**: **Angles between critical joints**, such as shoulders, elbows, and wrists, are computed, capturing **relative orientations** essential for pose assessment. As shown in equation(2)
- **Distance Measurement**: **Euclidean distances between select body parts**, such as wrists and ankles, are recorded, adding **spatial data** that complements the angle-based analysis. As shown in equation(1)

2.4.4 Data Collection and Compilation

For each image processed, the computed features are stored in a centralized dataset: Joint angles and distances are stored for each image in a structured format, enabling consistent retrieval for later analysis. The labels assigned to each pose allow for accurate classification and comparison. A tracking mechanism is implemented to record the number of images processed, ensuring data consistency and integrity across the dataset.

2.5. Model Architecture

The core of our classification system is a convolutional neural network (CNN) architecture, designed to learn and distinguish between complex yoga poses. Our model consists of several convolutional and pooling layers, followed by fully connected layers, which capture spatial hierarchies in the yoga poses.

The pose classification model is built using a neural network with a series of fully connected layers. The architecture begins with an input layer matching the dimensionality of the standardized data features. It includes two dense hidden layers with ReLU activation functions, along with dropout and batch normalization layers to enhance generalization and stability during training. The output layer employs a softmax activation to generate class probabilities for each yoga pose. The model is compiled with the Adam optimizer, categorical crossentropy as the loss function, and accuracy as the performance metric. Training is performed over 50 epochs with a validation split to monitor model performance. The model, upon completion of training, can be saved for subsequent use in classifying poses in real-time applications.

3. Experiments and Results

We conducted extensive experiments to validate the performance of our model on the yoga pose dataset. We varied parameters such as the number of epochs, batch size, and learning rate to optimize accuracy.

3.1. Evaluation Metrics

We used accuracy as the primary metric to evaluate the classification performance. Additionally, we analyzed the model's confusion matrix to understand specific pose classification performance and identify common misclassifications.

3.2. Visualization of Results

Figure **??** shows the confusion matrix for the classification results. This highlights the high accuracy of the model and shows which poses are most commonly misclassified.

Metric	Value	Description
Accuracy	71.2%	Overall classification
Training Accuracy	71.67%	Training dataset
Validation Accuracy	71.24%	Validation dataset

Table 1. Performance Metrics for Yoga Pose Classification Model.

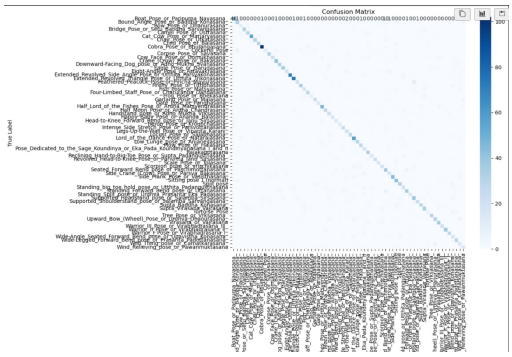

Figure 2. Confusion Matrix of Yoga Pose Classification Results.

4. Yoga Pose Classifier Web Application

This project involves the development of a user-friendly web application built using Anvil, which integrates our model designed for yoga pose classification. The primary objective of the web app is to allow users to upload images of themselves performing yoga poses, and the app automatically identifies and returns the name of the corresponding yoga pose.

5. Appendix

Below is the link to the code used for the Yogapose-claasification and the Dataset used respectively. The ppt for our project is here

6. Conclusion

This project demonstrates a successful approach to classifying yoga poses using a CNN model, achieving high accuracy and effectiveness within a web application. The Yoga Pose Classifier Web Application combines machine learning and web development to provide users with real-time pose identification, offering instant feedback on pose accuracy. This user-friendly tool enhances yoga practice, making it easier for users to improve their technique. Future work may involve expanding the model to recognize additional poses, optimizing it for mobile devices, and integrating video-based classification for real-time feedback.

References

[1] S. Kothari, "Yoga pose classification using deep learning," 2020. 4

[2] M. Verma, S. Kumawat, Y. Nakashima, and S. Raman, "Yoga-82: A new dataset for fine-grained classification of human poses," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020, pp. 4472–4479. 4

[3] S. Aggarwal, D. Saxena, and P. K. Jain, "Real-time yoga pose classification using deep learning and pose estimation," in *IEEE International Conference on Artificial Intelligence and Data Science*, 2023. 4

[4] Django, "Django: The web framework for perfectionists with deadlines," https://www.djangoproject.com, n.d., accessed: 2024-11-09. 4

[5] Flask, "Flask: A micro web framework written in python," https://flask.palletsprojects.com, n.d., accessed: 2024-11-09. 4

[6] S. Gupta, S. Sharma, and P. K. Sinha, "Yoga pose recognition using deep convolutional neural networks," in *Proceedings of the International Conference on Signal Processing, Communication, and Computing*, 2022. 4

[7] P. K. Sahu, A. R. P. Manikandan, and S. K. Gupta, "Yoga pose classification with wearable sensors using machine learning algorithms," *International Journal of Healthcare Engineering*, vol. 2021, 2021. 4

[8] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. Yong, J. Lee, W.-T. Chang, W. Hua, M. Georg, and M. Grundmann, "Mediapipe: A framework for building perception pipelines," 06 2019. 4

[1] [2] [3] [4] [5] [6] [7] [8]

7. Group Members

Kshitij Shukla
Roll - 230583

Tarawat Shreyansh Prakash
Roll - 231087

Ayushi Ojha
Roll - 231040607

Naman Patidar
Roll - 230679