An Improved Approach for Yoga Pose Estimation of Images

G Shirisha
School of Computer Science and Engineering
KLE Technological University
Hubballi, India
Email: 01fe21bcs157@kletech.ac.in

Neha R Bhat
School of Computer Science and Engineering
KLE Technological University
Hubballi, India
Email: 01fe21bcs194@kletech.ac.in

Akshata S Hamasagar
School of Computer Science and Engineering
KLE Technological University
Hubballi, India
Email:01fe21bcs165@kletech.ac.in

Ananya A Hosamani
School of Computer Science and Engineering
KLE Technological University
Hubballi, India
Email: 01fe21bcs048@kletech.ac.in

Priyadarshini Patil
School of Computer Science and Engineering
KLE Technological University
Hubballi, India
Email: priyadarshini.patil@kletech.ac.in

Meena S M
School of Computer Science and Engineering
KLE Technological University
Hubballi, India
Email:msm@kletech.ac.in

***Abstract*—This project focuses on advancing yoga pose estimation using deep learning. The dataset undergoes thorough preprocessing to ensure integrity, addressing issues like corrupted images. Drawing upon a pre-trained VGG16 network and custom layers, the model is tailored for yoga pose intricacies. The model is evaluated on both training exercises and real-world tests, guiding further enhancements. The trained model is saved for future use, and testing involves systematic validation on new images using a translation dictionary for result interpretation. Our proposed model has achieved the accuracy of 97.30% with a validation loss of 52.82% and validation accuracy of 98.82% for Yoga Set 1 and accuracy of 97.48% with a validation loss of 55.63% and validation accuracy of 97.72% for Yoga Set 2. We expanded pose prediction from three to six poses using models. The high accuracy achieved suggests potential applications in not only pose recognition but also in providing valuable insights for improving form and technique. The project's outcomes contribute to computer vision and pave the way for applications in real-time feedback systems, promising advancements in yoga practice enhancement.**

***Index Terms*—Yoga Pose Estimation, VGG16, Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Keypoint locations, Pose labels, Pixel values, Rectified Linear Unit(ReLU).**

I. Introduction

Estimating the human posture, a fundamental aspect of computer vision, involves accurately determining the main joints and limbs of an image. This process plays a pivotal role in decoding the spatial arrangement of the human body in a two-dimensional context. Identifying anatomically important points, such as the ankles, knees, hips, wrists, elbows, shoulders, and major points of the head, lays the foundation for a comprehensive understanding of human posture and movement dynamics. Various methodologies and techniques are used to solve the problem of estimating human posture. Traditional computer vision methods often rely on handmade features and algorithms, but modern approaches harness the power of deep learning. Convolutional Neural Networks (CNNs) have been proven effective at capturing hierarchical features in images to implement accurate joint localization. Recurrent Neural Networks (RNNs) are useful for capturing temporal dependencies in a series of poses, providing a more nuanced understanding of dynamic movements. The integration of these techniques has resulted in a hybrid model that provides a balance between accuracy and efficiency in estimating the human posture.

Furthermore, the advent of 3D pose estimation techniques has allowed for a more comprehensive understanding of human posture in three-dimensional space. Depth information from sensors like depth cameras or stereo cameras is leveraged to estimate the depth coordinates of joints, enabling the reconstruction of 3D skeletal poses. This additional dimension enriches the accuracy of posture estimation, offering valuable insights into the spatial relationships between body parts. Integrating 3D pose estimation into the hybrid model further refines its ability to analyze complex movements and provides a more holistic perspective on human posture dynamics.

The ongoing exploration of novel data augmentation techniques, regularization methods, and model architectures promises to further enhance the accuracy, generalization, and real-world applicability of human posture estimation systems.

II. Related Work

A. [1]Learning Delicate Local Representations for Multi-Person Pose Estimation.(15 Jul 2020)

- This paper has proposed a novel method named Residual Steps Network (RSN) for multi-person pose estimation.

RSN focuses on efficiently aggregating features to enhance the precision of keypoint localization.
- This paper introduces an attention mechanism known as Pose Refine Machine (PRM) to further refine keypoint locations. Through extensive experimentation, the approach demonstrates state-of-the-art performance on two benchmark datasets: COCO and MPII. Notably, the single model achieves an impressive accuracy of 78.6 on the COCO test-dev dataset and 93.0 on the MPII test dataset, establishing the efficacy of RSN and PRM in advancing the field of multi-person pose estimation.

B. [2]Human Pose Estimation Using Deep Learning: A Systematic Literature Review(2023)

- This paper provides insights into the selection process for candidate papers in human pose estimation studies, emphasizing the use of quality assessment criteria for transparency and objectivity. It delves into the utilization of diverse datasets for single and multipose estimation, including images and video frames with human body joint annotations. The document also discusses various loss functions, popular CNN architectures, and approaches for estimating single person's pose in images. Furthermore, it highlights challenges in estimating multiperson poses, such as occlusions and varying human sizes, which are ongoing concerns in the field of human pose estimation.

C. [3]Deep High-Resolution Representation Learning for Human Pose Estimation(CVPR-2019)

- The paper presents the HRNet architecture for human pose estimation, focusing on learning reliable high-resolution representations. The key aspects of the HRNet architecture include maintaining high-resolution representations throughout the process and fusing multi-resolution representations repeatedly. The future works include applying the HRNet to other dense prediction tasks and investigating the aggregation of multi-resolution representations in a less light way. The results on the MPII validation set and the PoseTrack dataset demonstrate the superior performance of the HRNet architecture in keypoint detection and video pose tracking.
- The HRNet-W32 achieves a 73.4 AP score, outperforming other methods with the same input size, and the HRNet-W48 also achieves high performance in multi-person pose tracking on the PoseTrack2017 test set.

D. [4]Implementation of Machine Learning Technique for Identification of Yoga Poses (2020)

- The study achieved an accuracy of 99.04% in identifying yoga poses using a Random Forest Classifier, demonstrating the effectiveness of machine learning techniques in yoga posture recognition.
- The dataset used in the study contained at least 5500 images of ten different yoga poses, and the tf-pose estimation Algorithm was utilized to draw a skeleton of a human body in real-time, enabling the extraction of angles of the joints for use as features in implementing various machine learning models.
- The research utilized six classification models of machine learning, including Logistic Regression, Random Forest, SVM, Decision Tree, Naive Bayes, and KNN, with different parameters, and achieved an overall accuracy of 94.28% across all machine learning models

III. Challenges

- Dataset Limitations: The effectiveness of deep learning models for human pose estimation is highly dependent on the quality and diversity of the training datasets. Limited or biased datasets may hinder the model's ability to generalize well to real-world scenarios.
- Complexity of Human Poses: Human poses can be highly intricate and dynamic, especially in activities involving various body movements. The model may struggle with accurately capturing and predicting complex poses, leading to potential performance issues.
- Real-time Performance Constraints: In applications where real-time processing is essential, the computational demands of the model may lead to latency issues. Ensuring that the model can provide accurate posture estimations in a timely manner, especially in dynamic scenarios, requires careful optimization of both the model architecture and the inference pipeline. Techniques like model quantization and efficient model architectures can be explored to address this challenge without compromising accuracy significantly.
- Generalization to Different Activities: The proposed model aims to balance accuracy and efficiency in estimating human posture. However, it may face difficulties in generalizing well across a wide range of activities, as certain poses and movements might be underrepresented or not adequately captured during training.
- Computational Resources: Deep learning models, especially those integrating CNNs and RNNs, can be computationally intensive. Ensuring access to sufficient computational resources for training and inference may pose a logistical challenge.
- Ethical Considerations: Human pose estimation systems may be used in various applications, including surveillance. Ethical considerations regarding privacy and potential misuse of the technology should be carefully addressed.

The main challenge we are addressing is to improve the accuracy of the model. Improving model accuracy involves optimizing data quality, augmenting features, and refining model architecture, while also considering techniques like transfer learning, ensemble methods, and hyperparameter tuning. Regular monitoring, interpretability, and continuous learning are crucial for maintaining and enhancing performance in dynamic environments.

IV. Proposed Methodology

- In this work, we investigated the feasibility of employing VGG16, a pre-trained convolutional neural network, for human yoga pose estimation. This architecture, originally designed for image classification, was leveraged for its strong feature extraction capabilities. We opted for transfer learning, replacing the final classification layer with new layers specific to our task of predicting keypoint locations or pose labels directly from input images. This approach allowed us to capitalize on the robust feature maps learned by VGG16 on a massive dataset, while modifying the model to efficiently capture the intricate spatial relationships and subtle variations inherent in human yoga poses. Through fine-tuning and hyperparameter optimization, we aimed to achieve accurate and efficient pose estimation performance, paving the way for the development of real-time feedback systems or pose classification tools to enhance yoga practice.

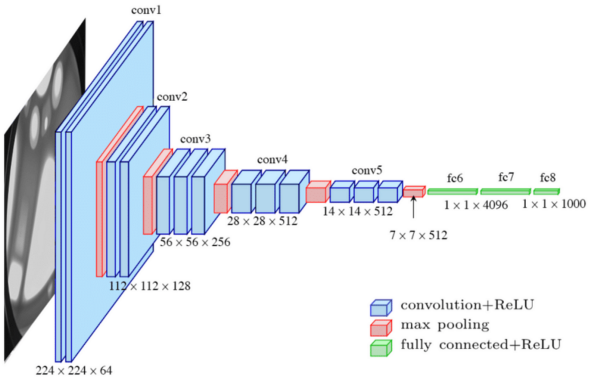

Fig. 1. Architecture of VGG16

- VGG16 is a fairly accurate model for image recognition tasks, and it has been shown to be effective for yoga pose estimation.
- VGG16 is a complex model with a large number of parameters. This can make it difficult to train and fine-tune, and it may require a lot of data to achieve good accuracy.

- We constructed our yoga pose estimation model upon the VGG16 architecture, leveraging pre-trained weights from ImageNet for transfer learning. The implementation involved the following steps:
  1. Loading Pre-trained Base: We initialized VGG16 with ImageNet weights, excluding its top classification layers, and set the input shape to 300x300x3 pixels to accommodate yoga pose images.
  2. Adding Custom Layers: We constructed a sequential model, starting with a flattening layer to transform the multidimensional feature maps from VGG16's base into a single vector. We then added a dense layer with 128 neurons activated by ReLU, followed by a dropout layer for regularization. Finally, we incorporated a final dense layer with 3 neurons (corresponding to our yoga pose classes), activated by a softmax function for probabilistic output.
  3. Combining Models: We integrated the base VGG16 model with our custom layers by defining a new model that takes the base model's input and outputs the predictions from our sequential model.
  4. Compiling the Model: We configured the model for training using stochastic gradient descent (SGD) with a learning rate of 0.0001, momentum of 0.9, and Nesterov momentum. We adopted categorical cross-entropy loss with label smoothing (0.2) to enhance generalization. Model performance was evaluated using accuracy as the primary metric.

V. Implementation

Implementation is done using VGG16, a pre-trained image detective, extracts intricate features from yoga poses. Its deep architecture dissects postures, revealing subtle nuances for accurate pose estimation, making it a powerful foundation for your system.

During implementation, modifications to VGG16 are made to align it with the specific requirements of yoga pose estimation. This involves adjusting the model's final layers to output predictions relevant to keypoint locations. The deep architecture of VGG16 allows for the extraction of abstract features, making it well-suited for accurate pose estimation.

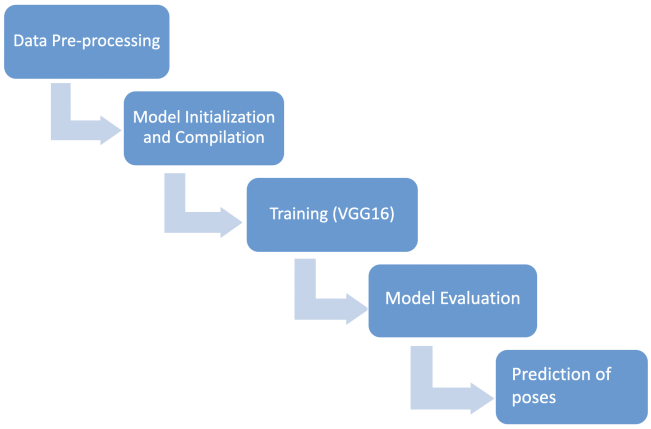

Fig. 2. Implementation Module

A. Dataset Collection and Input

We used a selected dataset of Yoga Pose Classification from 'Kaggle' [5] consisting of image data for 6 poses which are divided into 2 different sets, Yoga Set 1 and Yoga Set 2. Yoga Set 1 Contains images for Tree Pose, Warrior1 Pose and Downward Dog Pose. Yoga Set 2 Contains images for Mountain Pose, Warrior2 Pose, Goddess Pose.

The detailed data information is mentioned below in TABLE I.

TABLE I
DATASET INFORMATION

Poses	Training Images
Tree	371
Warrior1	432
Downward Dog	381
Warrior2	455
Mountain	453
Goddess	406

Fig. 3. Yoga Pose Dataset

B. Preprocessing

Prior to training our Yoga Pose Estimation Model, we meticulously cleansed the dataset to ensure its integrity and prevent potential errors during model training. This involved the following steps:

Identifying Corrupted Images: We employed the *removeCorruptedImages* function to systematically scan both training and validation image directories for files with structural or format issues. This proactive approach safeguards model training from encountering problematic data that could hinder learning or lead to unpredictable outcomes.

Verifying Image Integrity: Within the *removeCorruptedImages* function, we attempted to open each image using the Python Imaging Library and invoked its verify method. This crucial step ensures that each image is well-formatted, readable, and compatible with subsequent processing, maintaining data quality throughout model development.

Removing Faulty Files: If any image failed the verification process, indicating potential corruption, we promptly removed it from the dataset using *os.remove*. This preventative measure maintains dataset integrity and prevents corrupted images from compromising model learning.

Targeting Specific Pose Directories: To streamline the process, we applied *removeCorruptedImages* to directories containing images for specific yoga poses, including Tree, Downward Dog, WarriorI, Goddess, Mountain, and Warrior2. This targeted approach ensures data quality within each pose category.

By addressing corrupted images during preprocessing, we enhance model robustness and establish a reliable foundation for accurate yoga pose estimation. Addressing corrupted images during preprocessing is a crucial step to ensure the robustness and reliability of the model for accurate yoga pose estimation. Corrupted images may contain anomalies, artifacts, or distortions that can adversely affect the model's ability to learn meaningful features and patterns. To enhance model robustness, the preprocessing stage involves identifying and handling these corrupted images effectively.

C. Resizing Images

- After removing corrupted images to ensure data integrity and rescaling pixel values to the [0, 1] range for optimization, we implemented shearing, zooming, and horizontal flipping to enhance dataset variability and improve model generalization. The *ImageDataGenerator* was utilized for efficient loading and preprocessing of images in batches during both training and validation. Images were resized to 300x300 pixels to align with the model's input requirements.
- To effectively capture the details of yoga poses, we constructed a model architecture that leverages the strengths of the pre-trained VGG16 network while tailoring it to our specific task. We began by establishing a solid foundation using the pre-trained VGG16 model, known for its expertise in visual feature extraction. We strategically excluded its top classification layers, as they were originally designed for a different image domain. This allowed us to tap into its rich visual understanding gained from millions of images, while giving us the flexibility to customize its output for yoga pose estimation. We then carefully crafted a series of custom layers atop the VGG16 base to address the unique challenges of our task. First, we flattened the multi-dimensional feature maps extracted by VGG16 into a single vector, enabling further processing. This was followed by a dense layer containing 128 neurons, activated by the ReLU function, which introduced non-linearity for enhanced learning. To combat overfitting and promote generalization, we incorporated a dropout layer that randomly deactivated 20% of neurons during training. Finally, we introduced a final dense layer with 3 neurons, matching the number of yoga poses in our dataset, and activated it with a softmax function to produce probabilistic predictions for each pose class.
- To enhance the model's ability to generalize across a diverse range of yoga practitioners and conditions, we implemented data augmentation techniques during the training process. By applying random transformations such as rotations, translations, and zooms to the input images, we artificially increased the diversity of the training dataset. This not only helped the model become more robust to variations in body orientations and scales but also mitigated the risk of overfitting to specific instances, ultimately contributing to a more versatile and accurate yoga pose estimation system.

D. Training of the Model

- Training of the model has been implemented to both Yoga Set 1 and Yoga Set 2 for 3 poses in each dataset.
- We first established a training schedule of 6 epochs, ensuring ample exposure to the diverse yoga poses within our dataset. To balance computational efficiency and effective weight updates, we opted for a batch size of 16 images, allowing the model to learn incrementally from each batch.
- We then engaged the model.fit() function to commence training, prompting the model to engage in a series of tasks. The history object meticulously documented training and validation metrics across epochs, enabling us to track the model's learning trajectory and identify potential areas for refinement.

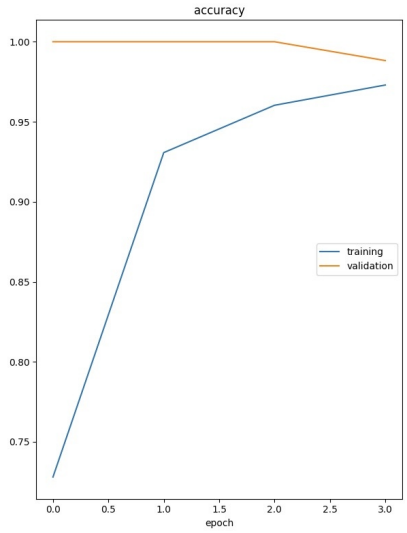

Fig. 4. Training and Validation Accuracy over Epochs of Yoga Set 1

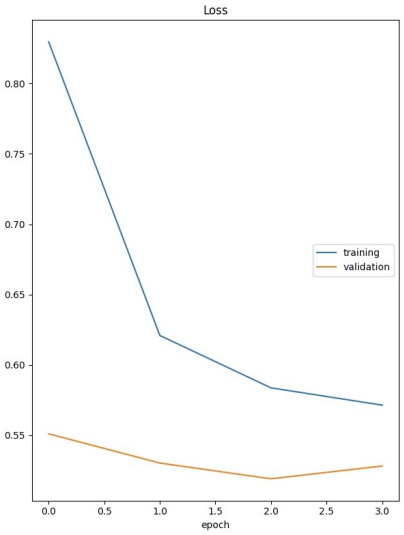

Fig. 5. Training and Validation Loss over Epochs of Yoga Set 1

- After training the model for Yoga Set 1, we achieved the accuracy of 97.30% with validation loss of 52.82% and validation accuracy of 98.82%.

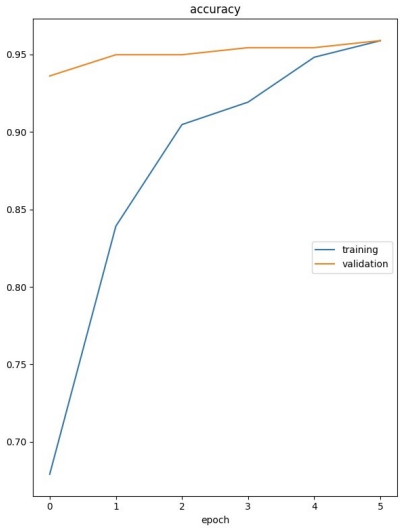

Fig. 6. Training and Validation Accuracy over Epochs of Yoga Set 2

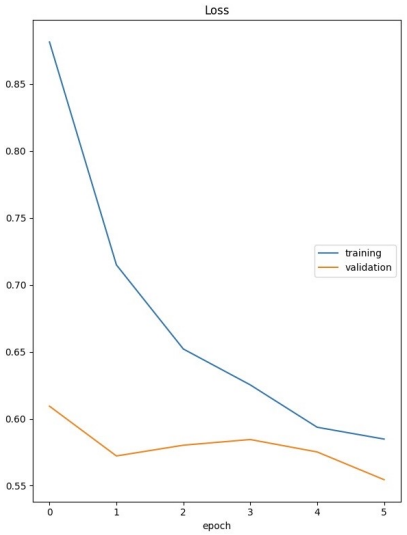

Fig. 7. Training and Validation Loss over Epochs of Yoga Set 2

- After training the model for Yoga Set 2, we achieved the accuracy of 97.48% with validation loss of 55.63% and validation accuracy of 97.72%.
- We tracked the progress of Yoga Set 1 and Yoga Set 2 on both training exercises (green/red lines) and real-world tests (blue/black lines) by measuring its mistakes (loss) and successes (accuracy). Plotting these metrics as colored lines lets us see how well it's learning and generalizing. Comparing training and validation lines reveals potential issues like overfitting, where it aces practice but mishandles real challenges. This dashboard is like a report card, helping us evaluate our model's strengths

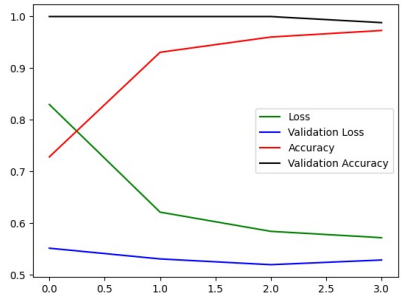

Fig. 8. Training & Validation Metrics for Model Performance of Yoga Set 1

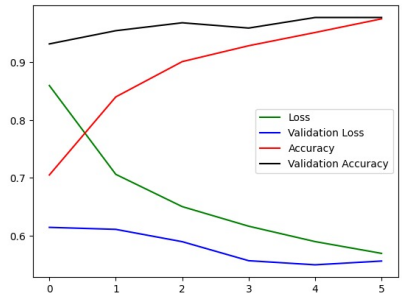

Fig. 9. Training & Validation Metrics for Model Performance of Yoga Set 2

and weaknesses, and guide further training decisions to ensure it excels in the real world.

VI. Results

- We move towards the process of saving the model, specifying a specific path for it.
- Before unleashing our trained model on new images, we first need to validate its performance. Therefore, at last, we move towards the process of testing the model. We define a dictionary(yoga_labels) that translates the model's numerical predictions (like 0, 1, or 2) into familiar yoga pose names like "downdog," "tree," or "warrior1." This translation process makes the model's output easily understandable.
- Our model needs the image of the yoga pose properly formatted before it can analyze it. *image.load_img(path, target_size=(300, 300))* resizes the image to the exact dimensions the model expects, and *image.img_to_array(img)* transforms it into a numerical format the model can interpret.
- We feed the prepared image to the model using loaded_model.predict(images, batch_size=10). This prompts the model to analyze the image and generates a list of probabilities for each possible yoga pose.
- We display the original image using plt.imshow(img). This allows us to visualize the pose alongside the model's prediction, providing a holistic understanding of the analysis.
- We now get the ultimate verdict on the pose. np.argmax(classes[0]) identifies the most likely pose by finding the highest probability in the list, and yoga_labels[pred_index] translates that numerical index back into the familiar pose name.

Fig. 10. Prediction Result 1

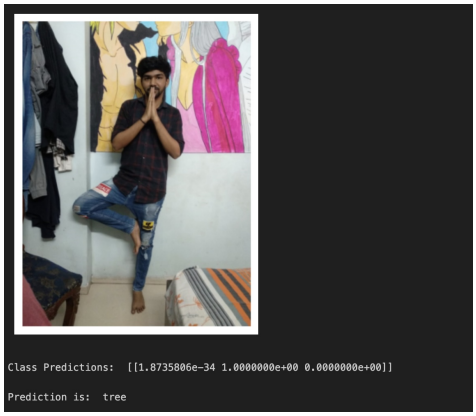

Fig. 11. Prediction Result 2

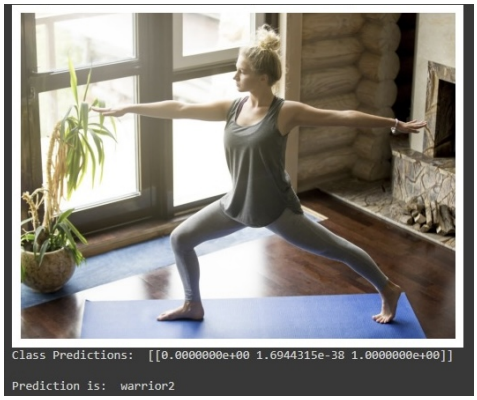

Fig. 12. Prediction Result 3

VII. Conclusion

In conclusion, this project focuses on the development and evaluation of model for yoga pose estimation using deep learning techniques. The key components of the project include data preprocessing, model construction, training, evaluation, and testing.

The preprocessing steps ensure the integrity of the dataset by addressing issues like corrupted images. The model architecture is constructed by leveraging the strengths of a pre-trained VGG16 network and incorporating custom layers to address the unique challenges of yoga pose estimation. The training process involves monitoring the model's progress through a dashboard, which aids in identifying potential issues such as overfitting.

The model is evaluated using training exercises and real-world tests for both the training datasets, providing insights into its strengths and weaknesses. This evaluation process guides further training decisions to enhance the model's performance in real-world scenarios.

The testing phase involves validating the model's performance on new images. A translation dictionary is utilized to make the model's numerical predictions easily understandable by associating them with familiar yoga pose names. The formatted images are fed into the model for analysis, and the results are visualized alongside the original images, providing a comprehensive understanding of the model's predictions.

The existing model achieved an impressive accuracy of 96.04% after 4 epochs for 3 poses, providing a foundation for further improvements. Our proposed model has achieved the accuracy of 97.30% with a validation loss of 52.82% and validation accuracy of 98.82% for Yoga Set 1 and accuracy of 97.48% with a validation loss of 55.63% and validation accuracy of 97.72% for Yoga Set 2 with 6 epochs for total of 6 poses. This signifies a noteworthy advancement in the model's predictive capabilities, particularly in real-world scenarios. The evaluation process played a pivotal role in guiding training decisions, leading to improved performance and robustness.

Overall, this project demonstrates a systematic approach to yoga pose estimation, encompassing data preparation, model development, training, evaluation, and testing. The integration of deep learning techniques, visualization tools, and careful model interpretation contributes to the creation of an effective system for understanding and predicting yoga poses. The project's outcomes pave the way for further advancements in real-time feedback systems and pose classification tools, with potential applications in enhancing the practice of yoga.

REFERENCES

[1] Cai, Y. et al. (2020). Learning Delicate Local Representations for Multi-person Pose Estimation. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, JM. (eds) Computer Vision – ECCV 2020. ECCV 2020. Lecture Notes in Computer Science(), vol 12348. Springer, Cham. https://doi.org/10.1007/978-3-030-58580-8_27

[2] Samkari, E.; Arif, M.; Alghamdi, M.; Al Ghamdi, M.A. Human Pose Estimation Using Deep Learning: A Systematic Literature Review. Mach. Learn. Knowl. Extr. 2023, 5, 1612-1659. https://doi.org/10.3390/make5040081

[3] arXiv:1902.09212 [cs.CV](or arXiv:1902.09212v1 [cs.CV] for this version)https://doi.org/10.48550/arXiv.1902.09212

[4] Y. Agrawal, Y. Shah and A. Sharma, "Implementation of Machine Learning Technique for Identification of Yoga Poses," 2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT), Gwalior, India, 2020, pp. 40-43, doi: 10.1109/CSNT48778.2020.9115758.

[5] Abhishek Gupta. (2020). Yoga Pose Classification, Version 1. Retrieved 2020 from https://www.kaggle.com/datasets/elysian01/yoga-pose-classification.