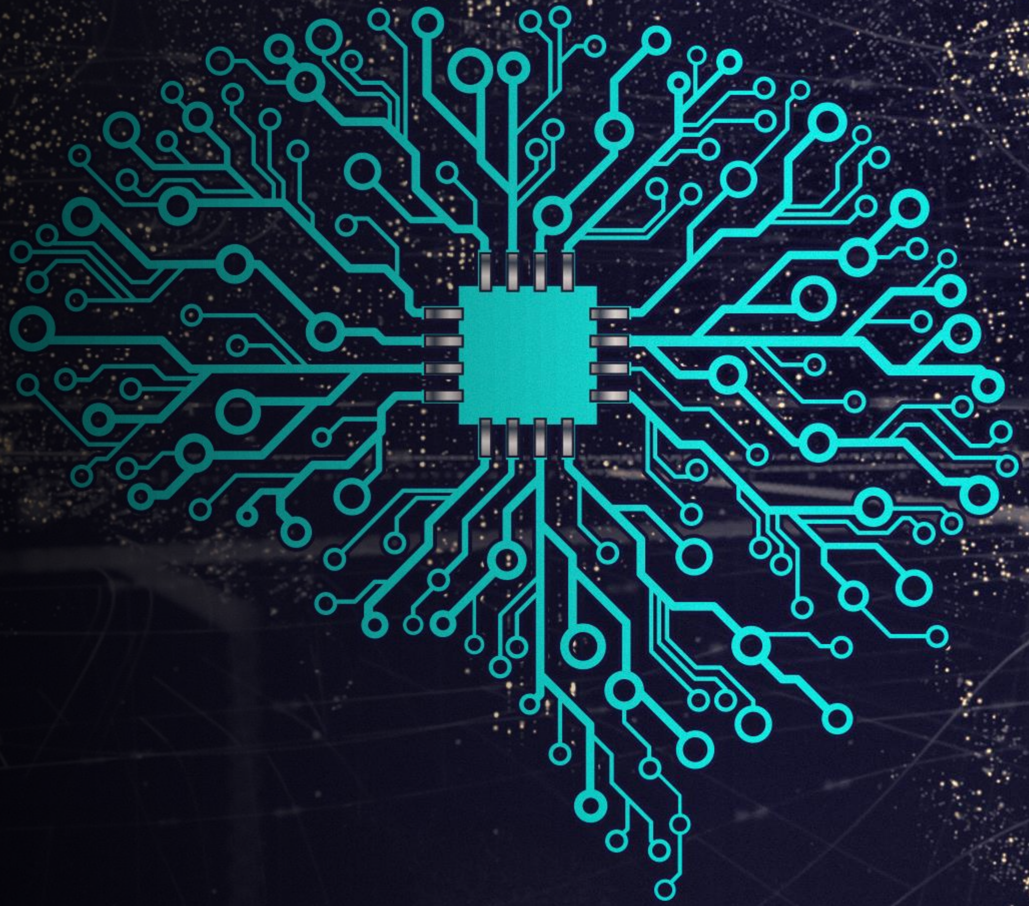
YOGA POSE CLASSIFICATION

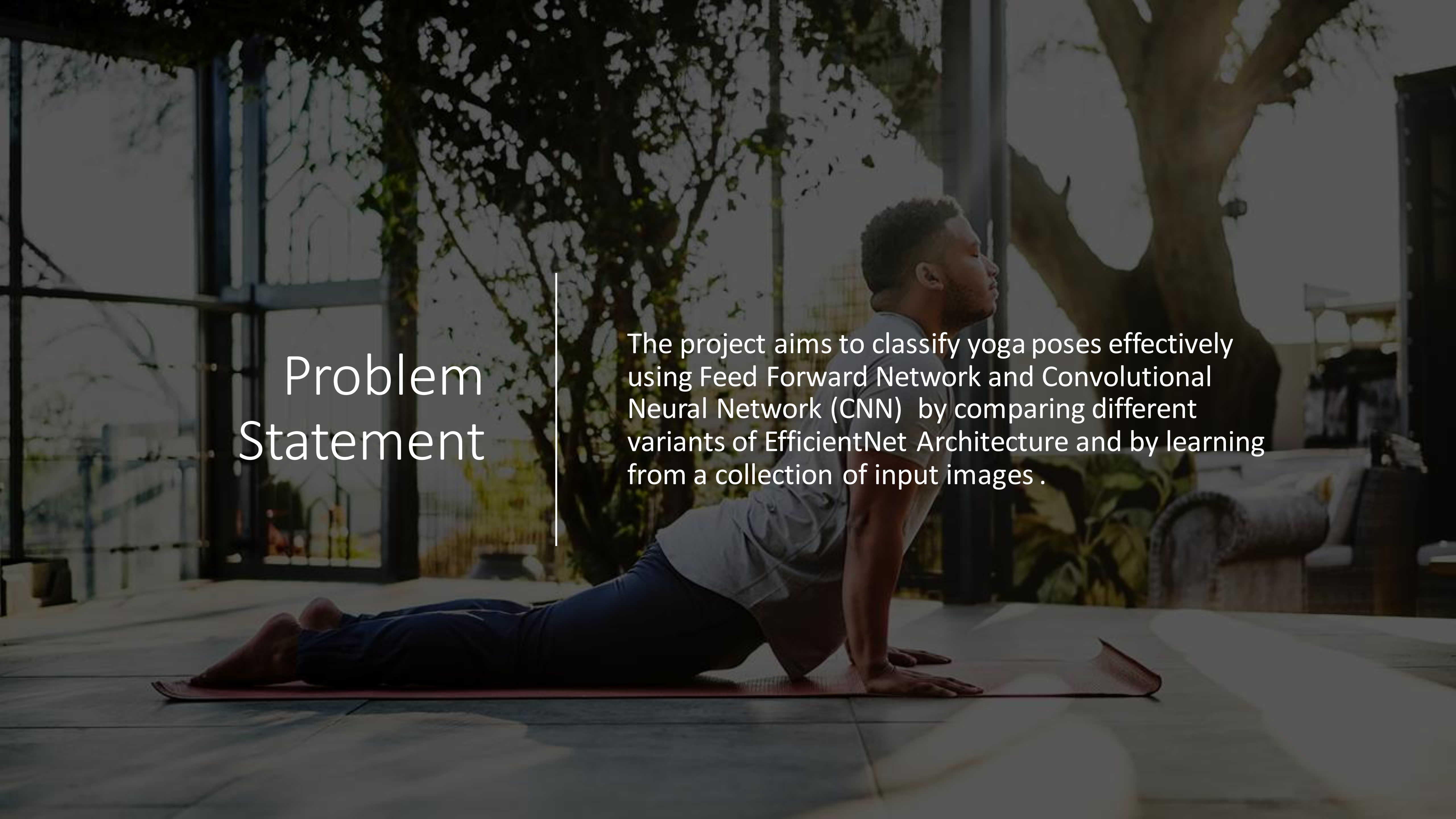
USING NEURAL NETWORKS



By
Anand K S [AM.EN.U4CSE19106]
Bharath Prathap Nair[AM.EN.U4CSE19113]
Rahan Manoj [AM.EN.U4CSE19144]

Background

- Yoga is a mind, body and spiritual enhancement practice developed traditionally in India which is now practiced all around the world due to its profound benefits.
- Today yoga has evolved so much that there are over 8,400,000 yoga poses. It can be noticed that many of these poses are too complex to be captured from a single point of view and is impossible for a human being to memorize and classify each of these yoga poses.
- Due to this, our project was developed so as to classify yoga poses more accurately by reducing the human error when perceived through the naked human eye.



Problem Statement

The project aims to classify yoga poses effectively using Feed Forward Network and Convolutional Neural Network (CNN) by comparing different variants of EfficientNet Architecture and by learning from a collection of input images .

RELATED WORKS

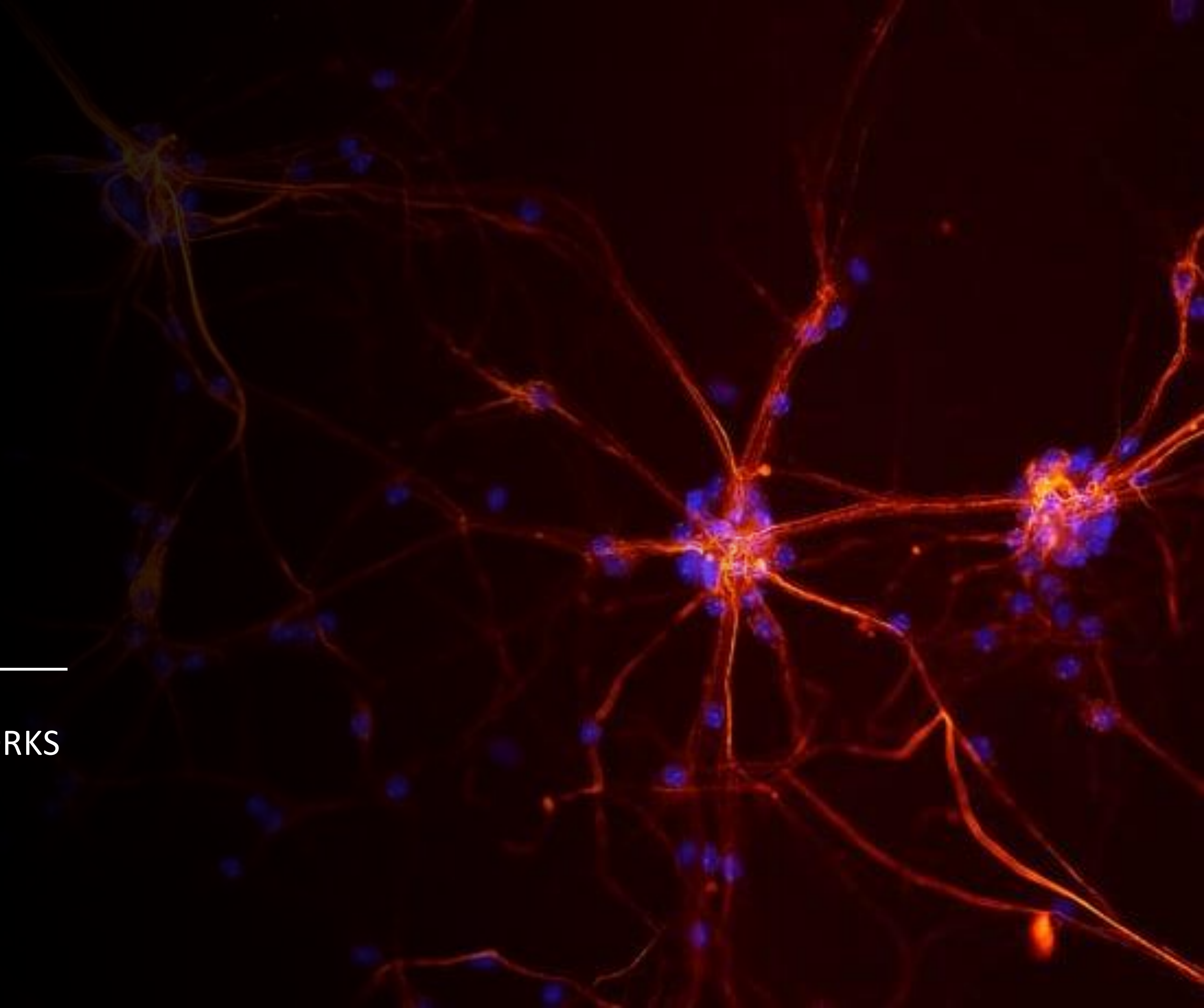
- **Kothari, Shruti, who had worked on``Yoga Pose Classification Using Deep Learning" has performed yoga pose estimation using keypoint detection methods such as OpenPose, PoseNet and PifPaf. It can also be found that the paper also focusses on Pose Classification using deep learning models such as Multi-layer Perceptron(MLP), Recurrent Neural Networks(RNN) and Convolutional Neural Network(CNN). This classification was done only for 6 yoga poses/classes. Included in Artificial Intelligence and Robotics Commons.**

- **In "Yoga-82: A New Dataset for Fine-grained Classification of Human Poses" published by Manisha Verma, Sudhakar Kumawat, Yuta Nakashima, Shanmuganathan , generated a new dataset named 'Yoga-82' and performed yoga pose classification using DenseNet Architecture, published in IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) 2020.**

- **E. Trejo, P. Yuan, "Recognition of yoga poses through an interactive system with kinect device" performed a yoga pose detection for 6 poses using Adaboost classifier and Kinetic sensors with really good accuracies, with the help of a depth sensor based camera, published in 2018 2nd International Conference on Robotics and Automation Sciences (ICRAS).**

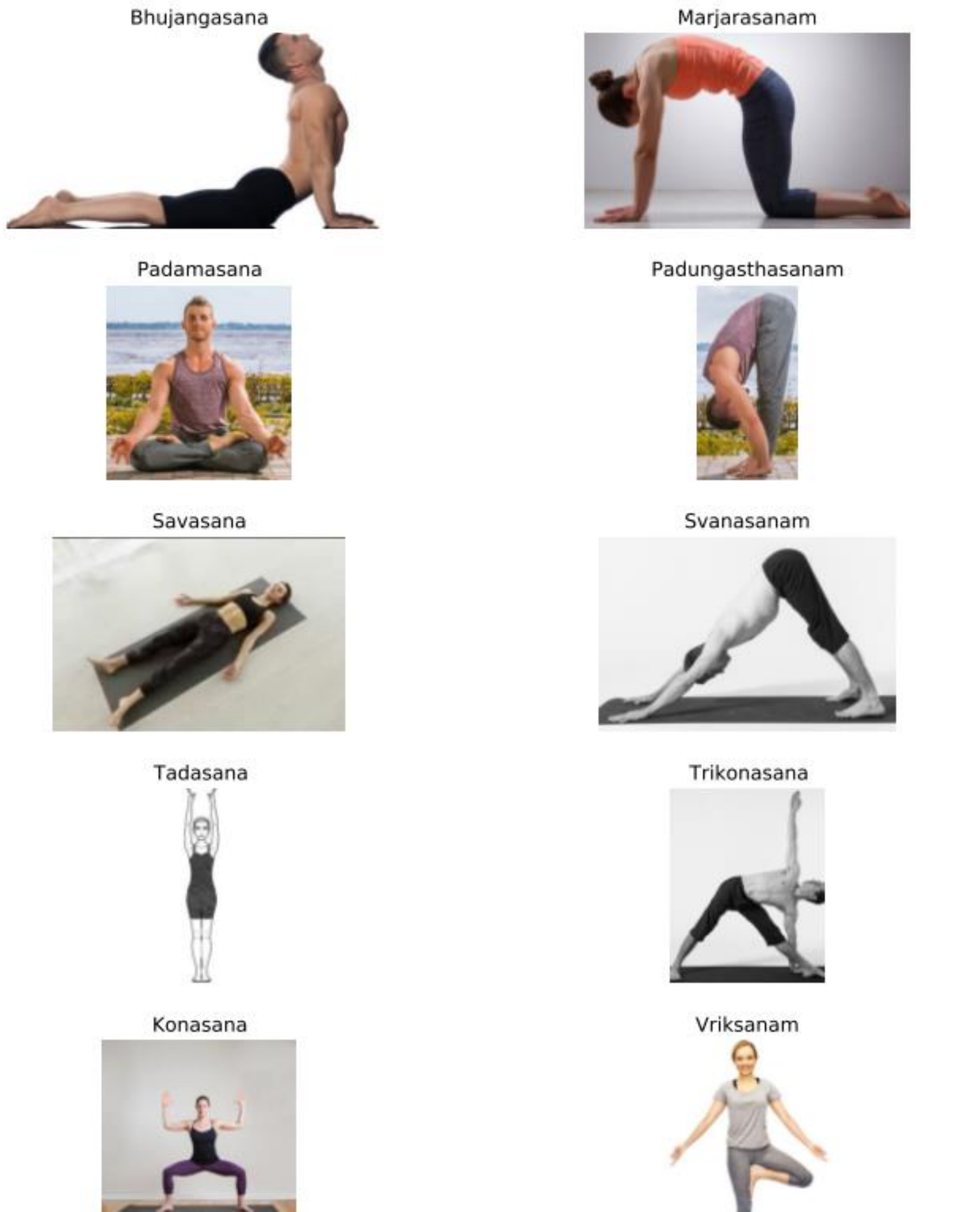
PHASE - 1

FEED-FORWARD NEURAL NETWORKS



DATASET

- Taken from Kaggle and from google images [using chrome extension].
- 1242 images.
- 10 classes/poses.
- Asanas : Svanasana, Padmasana, Bhujangasana, Utkata-konasanam, Marjarasanam, Trikonasana, Vriksasana, Padungasthasanam, Savasana and Tadasana



SOLUTION APPROACH

DATA PREPROCESSING

- Image size was set to 3 * 128 * 128
- Normalised the pixel values.
- Data Augmentation:
    - Rotate
    - Flip
    - Adding Noise
    - Blur
    - Changing Brightness and Contrast
    - Shearing
- Training data increased by 9 times.
- Batch size = 32

ARCHITECTURE DETAILS

- Input Layer dimension = 3 * 128 * 128
- First hidden layer dimension = 512
- Second hidden layer dimension = 1024
- Third hidden layer dimension = 512
- Output layer dimension = 10 [As we have 10 classes]

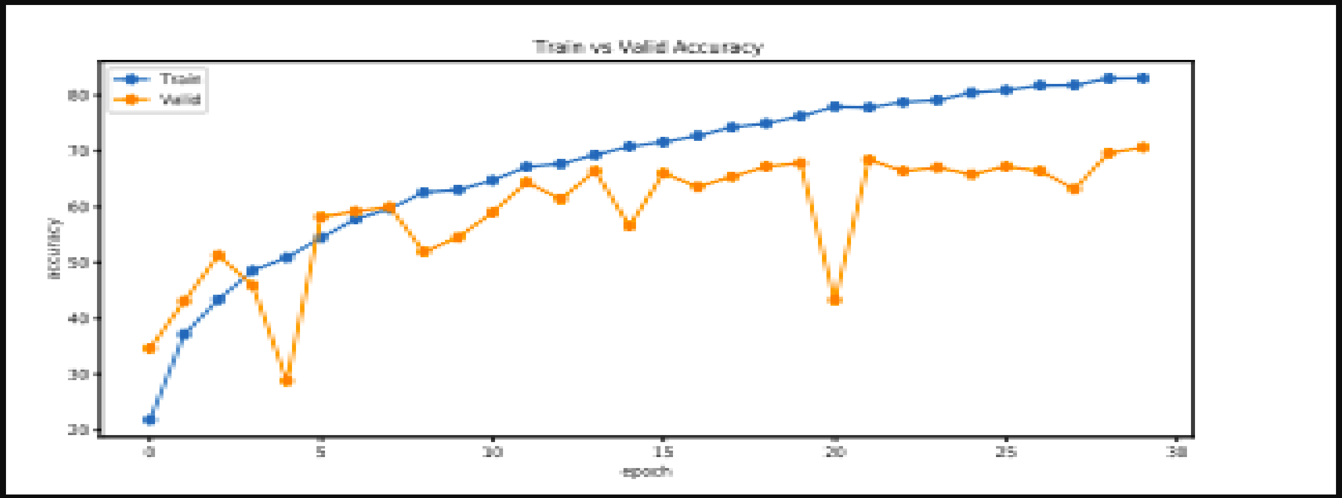
- Activation function of hidden layers = Leaky ReLU
- Activation function of output layers = LogSoftMax
- He initialisation
- Dropout = 50%
- Loss function = Negative Log Likelihood Loss
- Optimiser = Adagrad
- Weight decay [L2 regularisation] = 0.01

HYPER PARAMETER TUNINGS

- Varying the number of layers and its dimensions.
- Varying loss functions and optimisers.
- Data Augmentation [As mentioned in previous slide].
- He and Xe initialisations.
- Regularisation techniques
    - L2 regularisation
    - Dropout

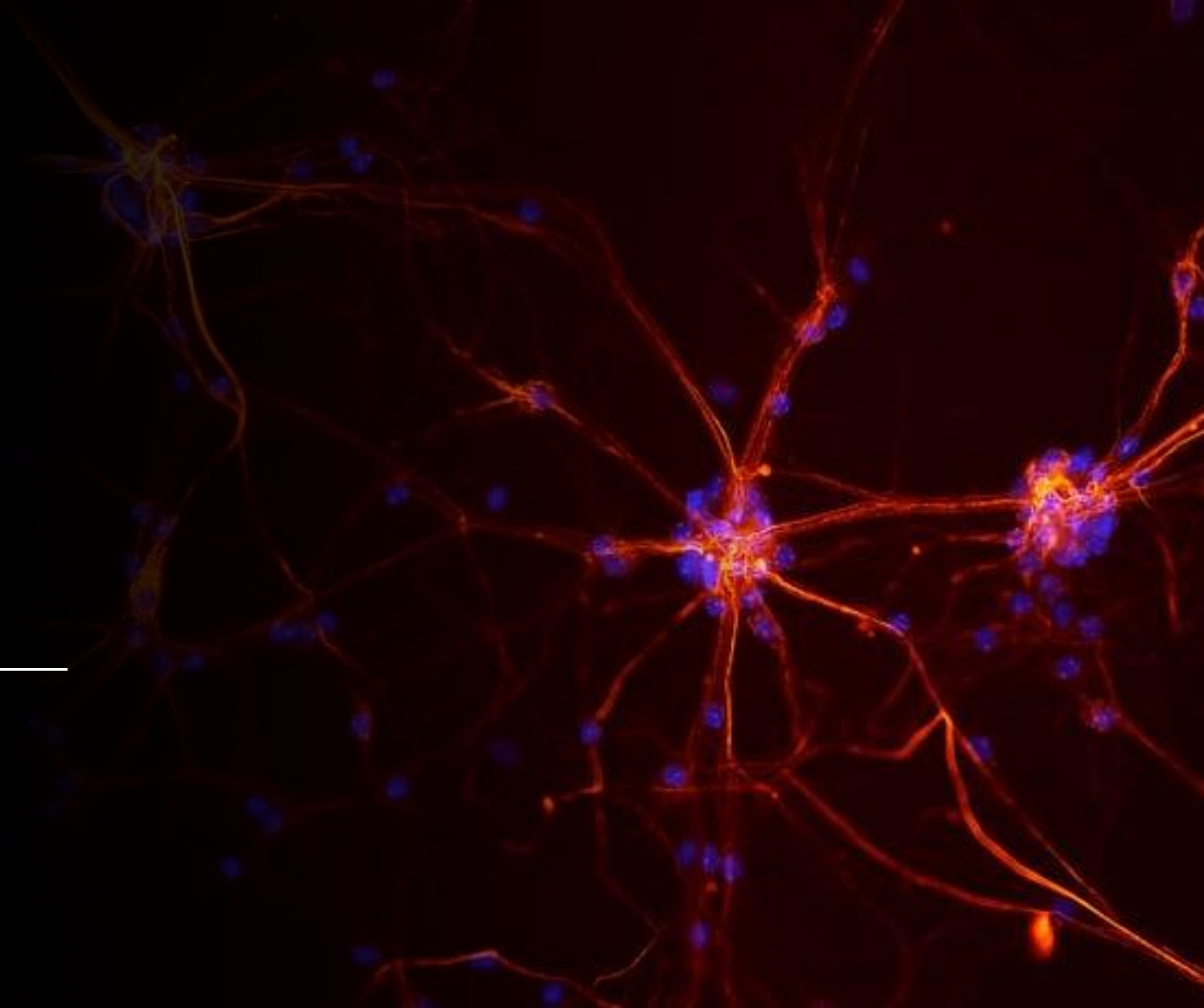
EXPERIMENTAL RESULTS

Sl.No	Optimiser	Loss Function	Accuracy
1.	Adagrad	Cross Entropy	63.343%
2.	AdamW	Cross Entropy	60.965%
3.	Adagrad	NLL Loss	70.623%
4.	AdamW	NLL Loss	65.392%



PHASE - 2

CONVOLUTIONAL NEURAL NETWORKS



DATASET

- Taken from Kaggle
- Dataset Link
- 5991 images.
- 107 classes/poses.

EFFICIENTNET

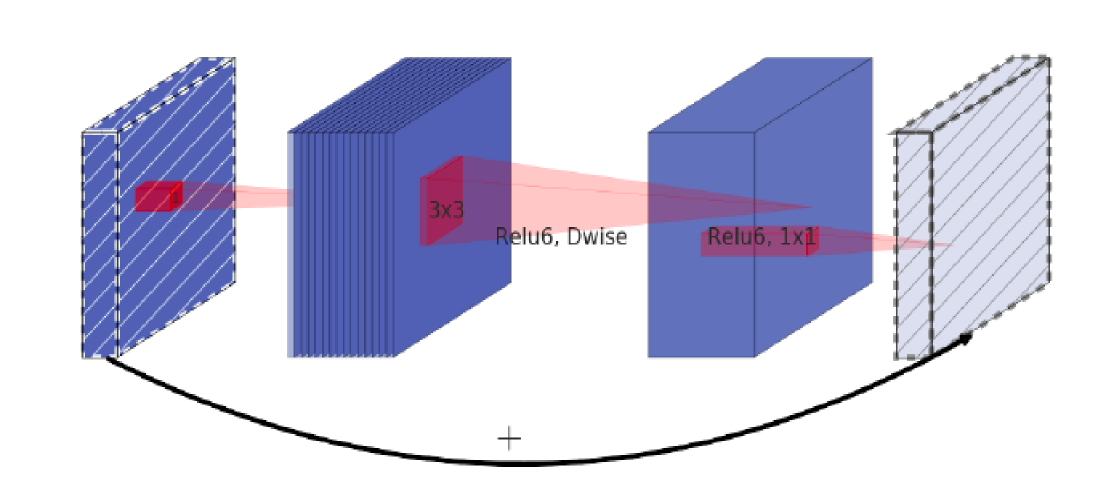
$$\text{depth: } d = \alpha^{\phi}$$
$$\text{width: } w = \beta^{\phi}$$
$$\text{resolution: } r = \gamma^{\phi}$$
$$\text{s.t. } \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$$
$$\alpha \geq 1, \beta \geq 1, \gamma \geq 1$$

- Dimensions like depth, width and resolution must be balanced during scaling.

- Basic bulding block of efficientnet is MbConv. [Mobile Inverted bottleneck convolution].
  - Inverted residual connections.
  - Squeeze and excitation.
  - Depthwise separable convolution.
    - Pointwise
    - Depthwise

- Uses SiLU activation function = x * sigmoid(x)

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224 × 224	32	1
2	MBConv1, k3x3	112 × 112	16	1
3	MBConv6, k3x3	112 × 112	24	2
4	MBConv6, k5x5	56 × 56	40	2
5	MBConv6, k3x3	28 × 28	80	3
6	MBConv6, k5x5	28 × 28	112	3
7	MBConv6, k5x5	14 × 14	192	4
8	MBConv6, k3x3	7 × 7	320	1
9	Conv1x1 & Pooling & FC	7 × 7	1280	1



Inverted Residual Block

SOLUTION APPROACH

DATA PREPROCESSING

- Image size was set to 3 * 224 * 224
- Normalised the pixel values.
- Data Augmentation:
    - Horizontal Flip
    - Adding Noise
    - Rotation
    - Blur
    - Translation
- Training data increased by 2 times.
- Batch size = 32

ARCHITECTURE DETAILS

- Efficientnet – Version : B0

- Input Layer : 3 * 224 * 224
- 237 layers of Efficientnet – B0
- Output layer dimension of EfficientNet-B0 – 1280
- Output dim of FC layer on top of pre-trained model = 107
- Output Activation function : LogSoftMax

- Loss function : Negative Log Likelihood Loss
- Optimiser : Adadelta

HYPER PARAMETER TUNINGS

- Added fully connected layer to the architecture.
- Tried different output activation functions.
- Varying loss functions and optimisers.
- Trained with different versions of EfficientNet.

EXPERIMENTAL RESULTS

Sl.No	Version	Optimiser	Loss function	Accuracy
1.	B0	Adadelta	NLL Loss	77.94%
2.	B1	Adagrad	Cross Entropy	76.06%
3.	B2	Adam	Cross Entropy	77.14%

Sl.No	Architecture	Accuracy
1.	VGG	66.89%
2.	ResNet	69.11%
3.	MobileNet	72.36%
4.	EfficientNet (Without tuning)	75.40%
5.	EfficientNet (With tuning)	77.94%

Comparison with existing

Kothari, Shruti, who had worked on ``Yoga Pose Classification Using Deep Learning" had performed yoga pose estimation only for **6 yoga poses/classes** whereas we have performed yoga pose classification on **107 classes**.

Similarly, E. Trejo, P. Yuan, in "Recognition of yoga poses through an interactive system with kinect device" performed a yoga pose detection for **6 poses** whereas we performed performed yoga pose classification on **107 classes .**

Contributions

- Different CNN architectures like VGG, ResNet, MobileNet and EfficientNet were compared and the best performance was observed with EfficientNet architecture which had an accuracy of 77.94%.
- Various EfficientNet variants such as EfficientNet-B0, EfficientNet-B1, EfficientNet-B2 were tested and it was observed that EfficientNet-B0 performs the best among them with an accuracy of 77.941% for the given dataset with Adadelta as the optimiser and Negative Log Likelihood Loss as the loss function.

References

- Ilya Loshchilov, Frank Hutter , "Decoupled Weight Decay Regularization," Published on 14 Nov 2017 (v1), last revised 4 Jan 2019 . https://arxiv.org/pdf/1711.05101.pdf
- Manisha Verma, Sudhakar Kumawat, Yuta Nakashima, Shanmuganathan Raman , "Yoga-82: A New Dataset for Fine-grained Classification of Human Poses" , https://arxiv.org/pdf/2004.10362.pdf
- Kothari, Shruti, "Yoga Pose Classification Using Deep Learning" (2020). Master's Projects. 932. DOI: https://doi.org/10.31979/etd.rkgu-pc9k
- Shubham Jain, April 19, 2018 - Analytics Vidhya, "An Overview of Regularization Techniques in Deep Learning", www.analyticsvidhya.com/blog/2018/04/fundamentals-deep-learningregularization-techniques/
- Shruti Saxena, "Yoga Pose Image classification dataset", www.kaggle.com/shrutisaxena/yoga-pose-image-classification-dataset

Thank You !