THE GEORGE WASHINGTON UNIVERSITY

WASHINGTON, DC

DATS 6303
Deep Learning

Human Action Recognition

Individual Project Report

Akshit Reddy Palle

1. Introduction

Human Action Recognition (HAR) focuses on identifying human activities from visual data such as images or video frames. It is a key component in applications like intelligent surveillance, sports analytics, human–computer interaction, assistive technologies, and behavioral monitoring.

In this project, I developed a complete deep-learning pipeline for image-based human action recognition using the EfficientNet-B1 architecture. The dataset contains 15 action classes (e.g., calling, clapping, cycling, drinking, eating, sleeping, running, using laptop, etc.), with 840 balanced images per category, making it suitable for supervised learning and transfer-learning approaches.

The complexity of the dataset arises from natural variations in pose, lighting, camera angles, background clutter, and inter-class similarities (for example: *sitting* vs *using laptop*, or *drinking* vs *eating*). To address these challenges, the model incorporates a modern training strategy using:

- **Aggressive image augmentations** to simulate real-world variability.
- **Mixup regularization** to improve generalization.
- **Label Smoothing and dropout** to reduce overfitting.
- **EfficientNet-B1 feature extractor** with a custom classification head.
- **Test-Time Augmentation (TTA)** for more stable final predictions

The overall pipeline consists of four components:

- Exploratory Data Analysis (EDA)
- Preprocessing & Augmentation
- Model Training
- Model Evaluation

2. Methodology

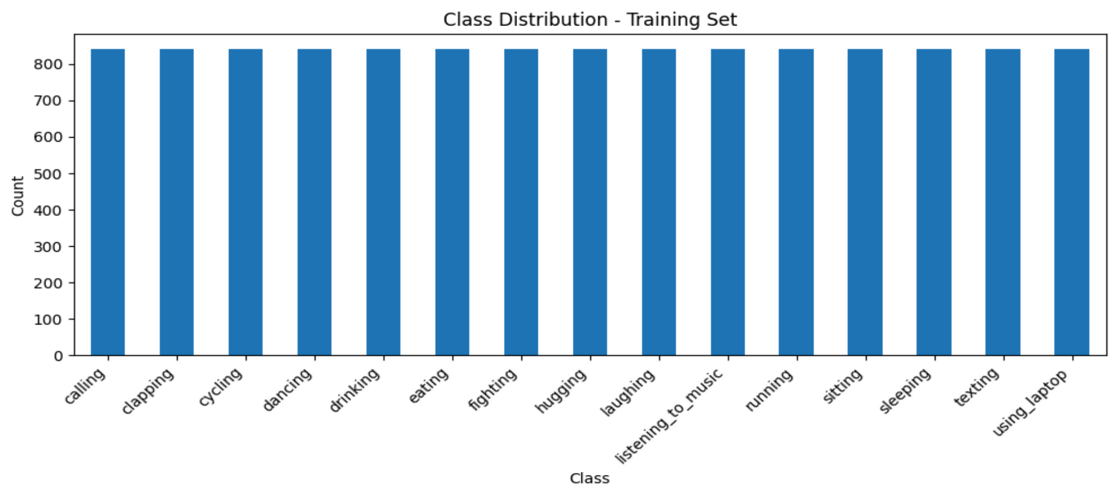
The development pipeline for the Human Action Recognition (HAR) system follows a structured and modular design, ensuring clear progression from raw data to final model evaluation. The methodology includes exploratory analysis, preprocessing, model construction, optimization, and evaluation.

I. Exploratory Data Analysis (EDA)

Before model development, the dataset was analyzed to understand its structure and validate its integrity. Key checks included:

A. Class distribution

All 15 action classes contained exactly 840 images, confirming perfect balance.



B. Sample Images Inspection

A set of random images was visualized to confirm action clarity and proper labeling.

These EDA steps ensure data integrity and reveal no missing files, corrupted samples, or class imbalances — providing a reliable foundation for model development.

II. Preprocessing and Data Augmentation

The images in the dataset vary in resolution and aspect ratio, so a consistent preprocessing pipeline was applied before training. All images were:

- resized and center-cropped to 240 × 240 pixels,
- converted to RGB tensors,
- normalized using standard ImageNet mean and standard deviation.

To improve generalization and reduce overfitting, I used strong data augmentation during training, including:

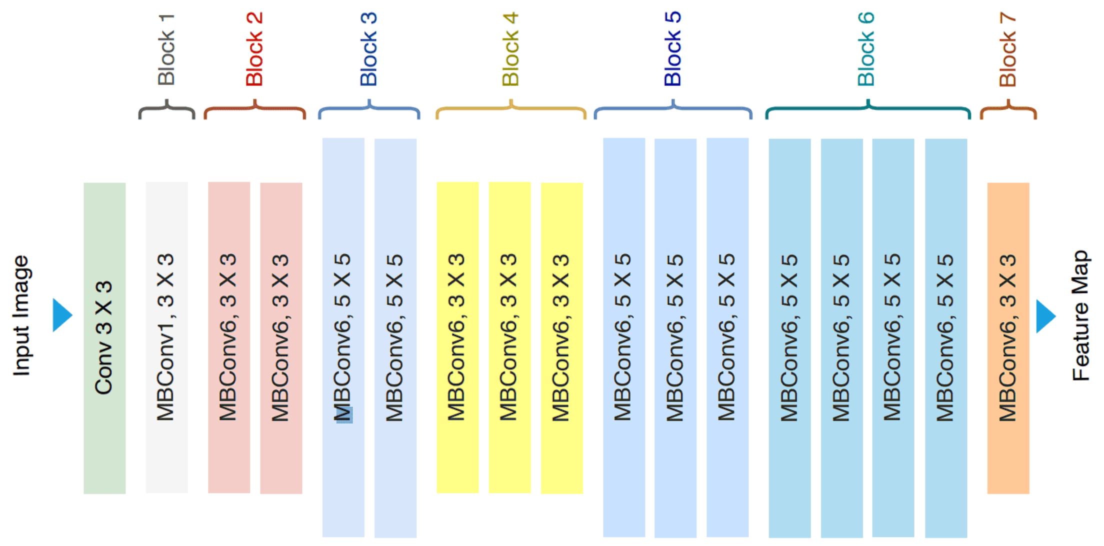
- random resized cropping,
- horizontal flips,
- small rotations and affine translations,
- slight perspective distortions, and
- color jittering (brightness, contrast, saturation, and hue).

In addition to these spatial augmentations, I also applied Mixup regularization with $\alpha = 0.2$, which linearly combines pairs of images and labels. This encourages the model to learn smoother decision boundaries and makes it less sensitive to noise. Overall, this preprocessing and augmentation pipeline helps the network handle variations in pose, lighting, background, and camera viewpoint.

III. Model Architecture

➔ Overview of Efficient Net

EfficientNet is a family of convolutional neural networks designed using compound scaling, a systematic method for scaling network depth, width, and input resolution in a balanced way. Instead of independently increasing only one dimension, EfficientNet introduces a unified scaling strategy that expands all three dimensions proportionally based on a set of fixed coefficients.



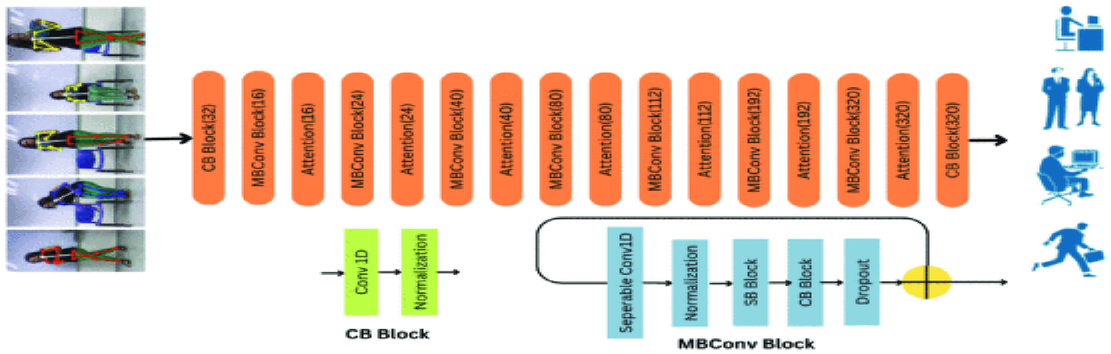
This approach allows the model to achieve high accuracy while maintaining computational efficiency. The architecture is built around Mobile Inverted Bottleneck Convolution (MBConv) blocks combined with Squeeze-and-Excitation (SE) attention modules. MBConv blocks capture spatial and contextual patterns efficiently through depthwise convolutions, while SE modules recalibrate channel-wise features to emphasize the most informative signals.

Overall, EfficientNet processes visual information through a structured flow:

- Stem convolution for low-level feature extraction.
- Stacked MBConv stages for hierarchical spatial–semantic learning.
- A classification head for global feature aggregation.

➔ EfficientNet-B1 Architecture Used in This Project

EfficientNet-B1 serves as the backbone for this Human Action Recognition system. It follows the core principles of the EfficientNet family—compound scaling and MBConv blocks—but introduces deeper network stages, more channels, and a larger input resolution (240 × 240) compared to EfficientNet-B0. These adjustments make B1 significantly more capable of capturing subtle variations in human posture, orientation, and context, which are essential for action recognition.



Internally, EfficientNet-B1 processes an input image through multiple stages:

- **Stem Convolution Layer**

A 3×3 convolution begins the pipeline by extracting low-level visual features such as edges, textures, and simple geometric structures.

- **Stacked MBConv Blocks (Stages 1–7)**

  Each MBConv block is composed of:

  - ➢ **1×1 expansion convolution**

    Expands channel dimensions to increase representation capacity.
  - ➢ **Depthwise convolution (kernel = 3 or 5)**

    Efficiently captures spatial patterns with minimal computational cost.
  - ➢ **Squeeze-and-Excitation (SE) attention**

    Recalibrates channel-wise feature responses by emphasizing the most informative activations.
  - ➢ **1×1 projection convolution**

    Compresses expanded features back into a compact representation.
  - ➢ **Skip connections**

    Facilitate gradient flow and preserve low-level details.

The stacked MBConv blocks progressively encode both local motion cues (limb direction, body tilt) and global structural patterns (overall pose), achieving strong representational power with remarkable efficiency.

- **Global Average Pooling (GAP)**

Compresses spatial features into a compact vector.

- **Classification Layer**

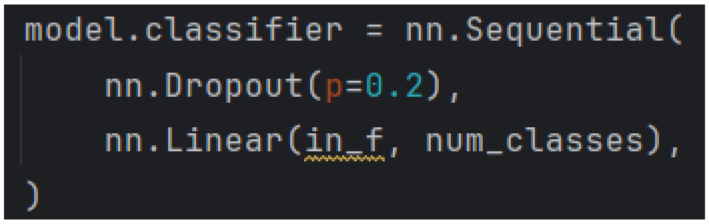
Outputs class probabilities across the 15 human action categories.

➔ Custom Modifications in This Project

To adapt EfficientNet-B1 for the specific demands of the HAR dataset, several targeted modifications were introduced. These modifications improve domain adaptation, regularization, and robustness.

- **Custom Classification Head**

The original classifier was replaced with:



```
model.classifier = nn.Sequential(
    nn.Dropout(p=0.2),
    nn.Linear(in_f, num_classes),
)
```

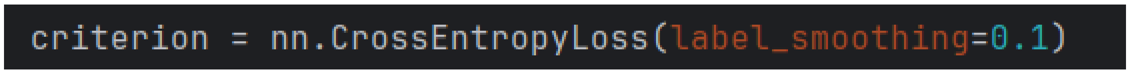
This provides:

(i). Dropout for reducing overfitting.
(ii). A new fully connected layer matching the dataset's 15 classes.

- **Label Smoothing**

HAR classes often share visual similarities (e.g., sitting vs. using laptop, calling vs. texting).

To prevent the model from becoming overly confident, label smoothing was applied:



```
criterion = nn.CrossEntropyLoss(label_smoothing=0.1)
```

This encourages the network to maintain softer, more generalizable decision boundaries.

- **Strong Data Augmentation Pipeline**

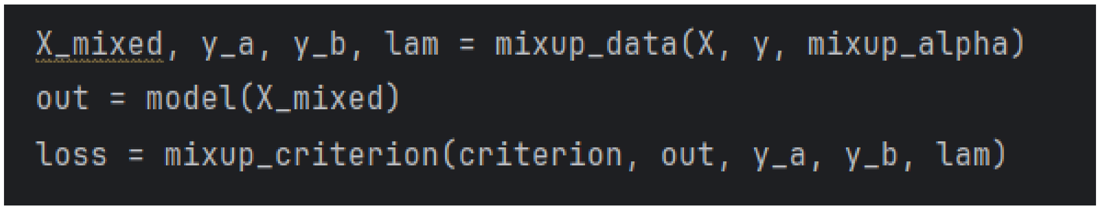
To simulate real-world variability, a rich augmentation set was used:

- ➢ RandomResizedCrop
- ➢ Horizontal Flip
- ➢ Rotation
- ➢ Affine transformations
- ➢ Perspective distortion
- ➢ Color Jitter

These augmentations introduce variations in lighting, pose, viewpoint, and background—critical for robust action recognition.

- **Mixup Regularization**

Mixup ($\alpha = 0.2$) blends images and labels



```
X_mixed, y_a, y_b, lam = mixup_data(X, y, mixup_alpha)
out = model(X_mixed)
loss = mixup_criterion(criterion, out, y_a, y_b, lam)
```

This reduces memorization, stabilizes training, and enhances generalization.

- **Gradual Unfreezing for Stable Fine -Tuning**

A two-phase transfer learning strategy was adopted:

➢ ***Stage 1*** *— Backbone Frozen*

Only the classifier trains. This allows EfficientNet's ImageNet features to adapt smoothly to HAR.

➢ ***Stage 2*** *— Full Network Unfrozen*

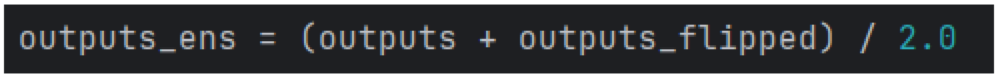
The entire backbone is fine-tuned with a lower learning rate.

This preserves pretrained knowledge while refining deeper layers for HAR-specific patterns.

- **Test Time Augmentation**

During inference, predictions are averaged across:

➢ The original image
➢ A horizontally flipped version



```
outputs_ens = (outputs + outputs_flipped) / 2.0
```

This produces more stable predictions, especially for actions that appear symmetric (e.g., clapping, hugging).

IV. Hyperparameter Tuning

The model was trained using a fixed set of hyperparameters selected through iterative experimentation. Key configurations include:

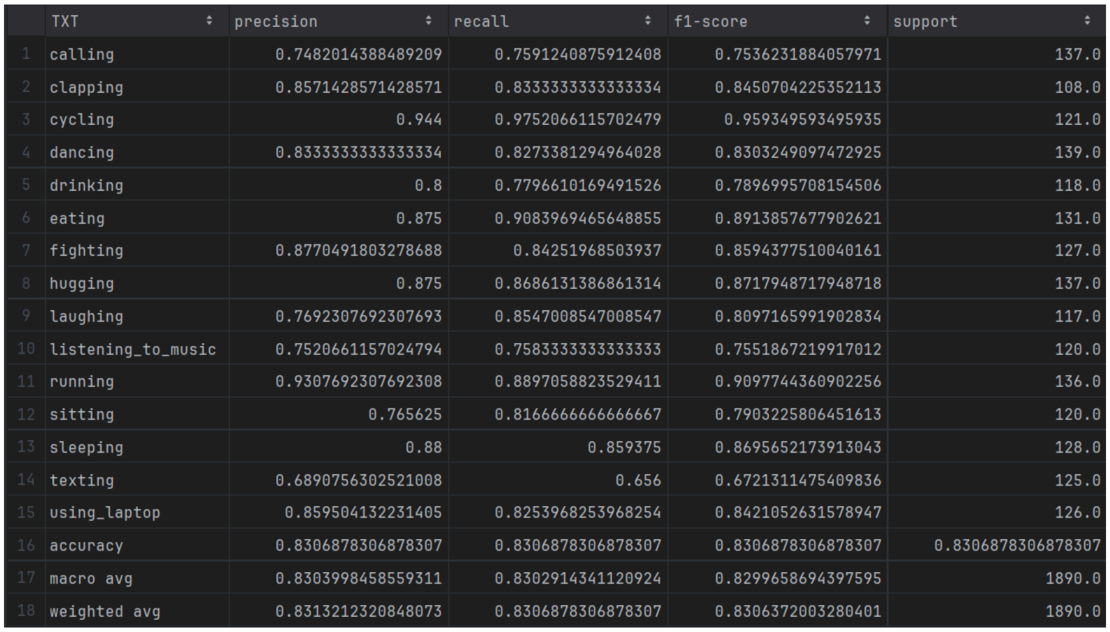
- Input Size: **240x240**
- Batch Size: **64**
- Optimizer: **Adam**
- Initial Learning Rate: **1e-4**
- Fine-tuning learning rate: **3e-5**
- Epochs: **25**
- Loss-Function: **Cross-entropy**
- Weight-Decay = **1e-4**

A ReduceLROnPlateau scheduler was used to lower the learning rate when validation loss stopped improving. The final model checkpoint was selected based on the highest validation accuracy.

V. Results

The final EfficientNet-B1 model achieved strong performance on the Human Action Recognition dataset. After fine-tuning with mixup, label smoothing, and strong augmentation, the model showed stable convergence with consistent improvement in both validation and test accuracy.

- Classification Performance:



	TXT	precision	recall	f1-score	support
1	calling	0.7482014388489209	0.7591240875912408	0.7536231884057971	137.0
2	clapping	0.8571428571428571	0.8333333333333334	0.8450704225352113	108.0
3	cycling	0.944	0.9752066115702479	0.959349593495935	121.0
4	dancing	0.8333333333333334	0.8273381294964028	0.8303249097472925	139.0
5	drinking	0.8	0.7796610169491526	0.7896995708154506	118.0
6	eating	0.875	0.9083969465648855	0.8913857677902621	131.0
7	fighting	0.8770491803278688	0.84251968503937	0.8594377510040161	127.0
8	hugging	0.875	0.8686131386861314	0.8717948717948718	137.0
9	laughing	0.7692307692307693	0.8547008547008547	0.8097165991902834	117.0
10	listening_to_music	0.7520661157024794	0.7583333333333333	0.7551867219917012	120.0
11	running	0.9307692307692308	0.8897058823529411	0.9097744360902256	136.0
12	sitting	0.765625	0.8166666666666667	0.7903225806451613	120.0
13	sleeping	0.88	0.859375	0.8695652173913043	128.0
14	texting	0.6890756302521008	0.656	0.6721311475409836	125.0
15	using_laptop	0.859504132231405	0.8253968253968254	0.8421052631578947	126.0
16	accuracy	0.8306878306878307	0.8306878306878307	0.8306878306878307	0.8306878306878307
17	macro avg	0.8303998458559311	0.8302914341120924	0.8299658694397595	1890.0
18	weighted avg	0.8313212320848073	0.8306878306878307	0.8306372003280401	1890.0

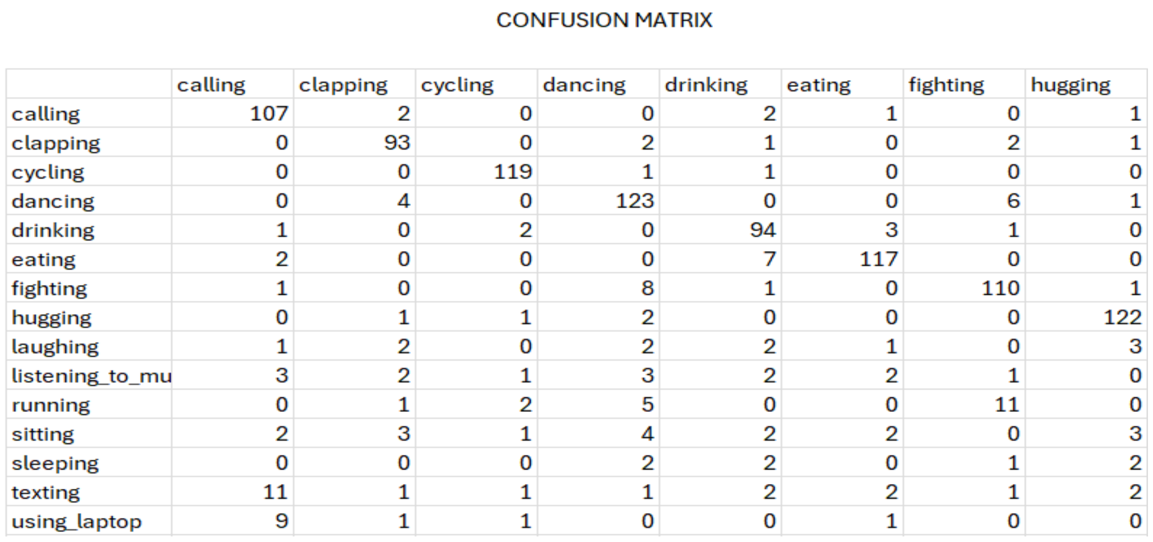
➢ Actions such as cycling, running, sleeping, and fighting achieve some of the highest F1-scores, indicating clear and distinguishable patterns in these classes.

➢ Actions with very similar visual appearance—such as calling vs. texting or sitting vs. using laptop—show slightly lower scores, which is expected because these categories share overlapping poses and hand positions.

➢ The overall performance (~83% accuracy and balanced F1-scores) demonstrates that the model is able to recognize a wide variety of human actions reliably.

These results confirm that the trained model has learned meaningful visual features and can differentiate between complex human activities effectively.

- Confusion Matrix



CONFUSION MATRIX

	calling	clapping	cycling	dancing	drinking	eating	fighting	hugging
calling	107	2	0	0	2	1	0	1
clapping	0	93	0	2	1	0	2	1
cycling	0	0	119	1	1	0	0	0
dancing	0	4	0	123	0	0	6	1
drinking	1	0	2	0	94	3	1	0
eating	2	0	0	0	7	117	0	0
fighting	1	0	0	8	1	0	110	1
hugging	0	1	1	2	0	0	0	122
laughing	1	2	0	2	2	1	0	3
listening_to_mu	3	2	1	3	2	2	1	0
running	0	1	2	5	0	0	11	0
sitting	2	3	1	4	2	2	0	3
sleeping	0	0	0	2	2	0	1	2
texting	11	1	1	1	2	2	1	2
using_laptop	9	1	1	0	0	1	0	0

➢ The strong diagonal pattern shows that the model correctly identifies most samples across all categories.

➢ Misclassifications mostly occur between actions that are naturally similar:

➔ Calling and Texting
➔ Sitting and Using Laptop

➢ Actions with distinctive full-body movement show minimal confusion

The confusion matrix helps illustrate where the model is confident and where visual similarity between actions leads to overlap.

- Overall Performance

- ➢ **Accuracy:** 0.83
- ➢ **F1 Macro:** 0.83
- ➢ **Weighted F1:** 0.83

VI. Summary and Conclusions

This project built a Human Action Recognition (HAR) system using the EfficientNet-B1 deep learning model. The work included exploring the dataset, preparing the images, training the model, and evaluating its performance.

A few important learnings from the project:

- The dataset was perfectly balanced, with 840 images for each of the 15 actions. This made training more stable.
- EfficientNet-B1 worked very well for this task. It was able to learn both small details (like hand positions) and bigger body movements.
- Techniques like data augmentation, label smoothing, mixup, and dropout helped the model avoid overfitting.
- Using gradual unfreezing allowed the model to start by learning the simple parts first and then fine-tune deeper layers slowly.

The final model reached about 83% accuracy and 83% F1-score, which is strong performance for a single-frame action recognition task. Actions with clear movement (like *cycling* and *running*) were recognized very well, while visually similar actions (like *calling vs. texting*) were more challenging.

➢ *Future Improvements*

There are a few ways this project could be improved in the future. One option is to train the model on higher-resolution images so it can pick up small details, like hand movements in actions such as texting or calling. We could also add more types of data augmentation to help the model handle different lighting, backgrounds, and camera angles. Trying larger versions of EfficientNet (like B2 or B3) or other modern models might also boost performance.

Overall, the project successfully developed an effective Human Action Recognition system using the EfficientNet-B1 architecture. Through careful preprocessing, strong regularization strategies, and a well-designed training pipeline, the model achieved consistent and reliable performance across all 15 action categories. The results show that EfficientNet-B1 is well-suited for still-image action recognition, and the final system provides a solid foundation that can be extended with higher-resolution inputs, additional augmentations, or temporal modeling for further improvement.

VII. References

1. https://ieeexplore.ieee.org/document/10344070
2. https://github.com/huggingface/pytorch-image-models
3. Nagadia, M. (2020). *Human Action Recognition (HAR) Dataset*. Kaggle. https://www.kaggle.com/
4. https://ieeexplore.ieee.org/document/8099678
5. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks: https://arxiv.org/abs/1905.11946
6. Geeks for geeks: https://www.geeksforgeeks.org/computer-vision/efficientnet-architecture/

VIII. Percentage of Code

Lines of code from Internet : 300
Lines of code modified: 70

Lines of code added: 210

Contribution: 45%