Vidyayāmruthamashnuthe

# B. N. M. Institute of Technology

An Autonomous Institution under VTU
Approved by AICTE, Affiliated to VTU

## Internship Phase-II
## Project Assessment Report
## Course: Deep Learning and Reinforcement Learning

## "Transfer Learning For Flower Classification"

### Team Members :


| NAME | USN |
|---|---|
| Sanchita H | 1BG23CS131 |
| Srusti M | 1BG23CS159 |


### Internship Offered By:
**E2E Management Services**

E2E
END TO END SOLUTIONS

# Project Objectives

The primary objective of this project is to **design and implement an intelligent image classification system** that accurately identifies the species of flowers using deep learning techniques. By leveraging the tf_flowers dataset and harnessing the power of **transfer learning with MobileNetV2**, the project aims to deliver a model that is not only accurate but also optimized for computational efficiency.

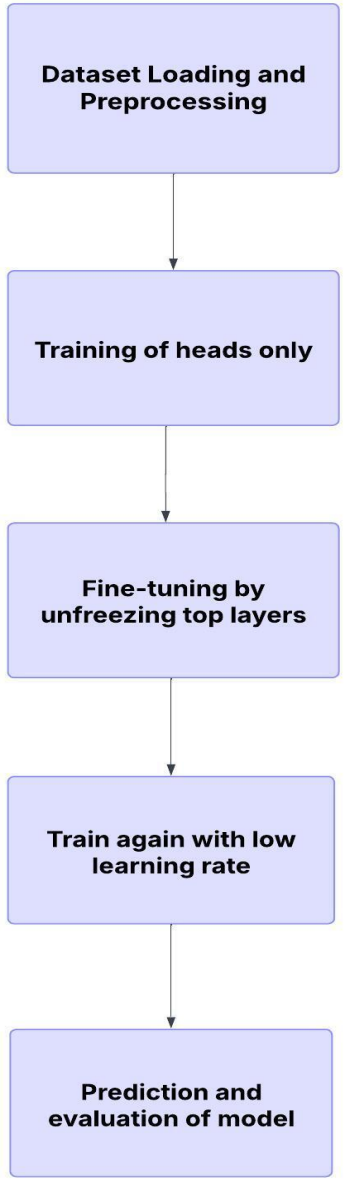
This project goes beyond traditional classification tasks by:

- Applying **fine-tuned transfer learning** to reuse pre-trained knowledge while adapting to the unique patterns in the flower dataset.
- Integrating a **custom loss function (Focal Loss)** to effectively handle class imbalances and improve sensitivity toward minority classes.
- Utilizing a **custom SGD optimizer** to gain better control over convergence and performance.
- Visualizing and interpreting the model's predictions to ensure explainability and class-wise performance insights.

# Methodology and Workflow

1. **Dataset:** Tf_flowers from Tensorflow datasets.
2. **Split:** Data is split into two parts:
   - 80% training
   - 20% validation
3. **Preprocessing:**
   - Resized images to 224 x 224
   - Normalized pixel values to [0,1]
4. **Model:**
   - MobileNetV2 pretrained on ImageNet
5. **Custom Optimizer:**
   - Implemented **MySGDOptimizer**
6. **Fine-Tuning:**
   - Unfroze top layers of base model

Flowchart:



# Key Assumptions

1. **MobileNetV2** pretrained on **ImageNet** is transferable to flower dataset.
2. **Tf_flower** dataset is sufficiently **balanced** for validation
3. **Frozen early** layers retain useful **general features**.
4. **Custom optimizer** mimics behaviour of **SGD** effectively
5. **Fine-tuning** with a smaller learning rate **enhances** performance.

# Model Evaluation and Analysis

**Model Architecture:**

- **Base Model**: MobileNetV2 (pre-trained on ImageNet, 16M parameters)
- **Custom Layers**:
  - GlobalAveragePooling → Dense(256, swish) → BatchNorm → Dropout
  - Dense(128, swish) → BatchNorm → Dropout
  - Dense(64, relu) → Dropout → Dense(5, softmax)
- **Loss Function**: Custom Focal Loss (handles class imbalance)
- **Optimizer**: Custom SGD Optimizer (MySGDOptimizer)
- **Training Phases**:
  - **Phase 1**: Freeze base model, train only head layers (5 epochs)
  - **Phase 2**: Unfreeze last 30 layers, fine-tune entire model (5 epochs)

**Evaluation Results:**

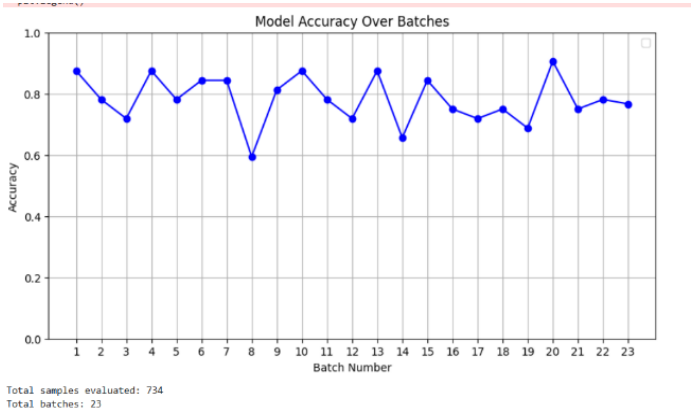
- **Phase 1 Accuracy** (Frozen Base): ~80.25
- **Phase 2 Accuracy** (Fine-tuned): ~96.47
- **Loss**: Decreased significantly after fine-tuning.
- **Per-Class Accuracy**: (From bar graph, you can add)
  - Rose: 96.45%
  - Sunflower: 83.58%
  - Dandelion: 96%
  - Tulip: 92.72%
  - Daisy: 90.34%

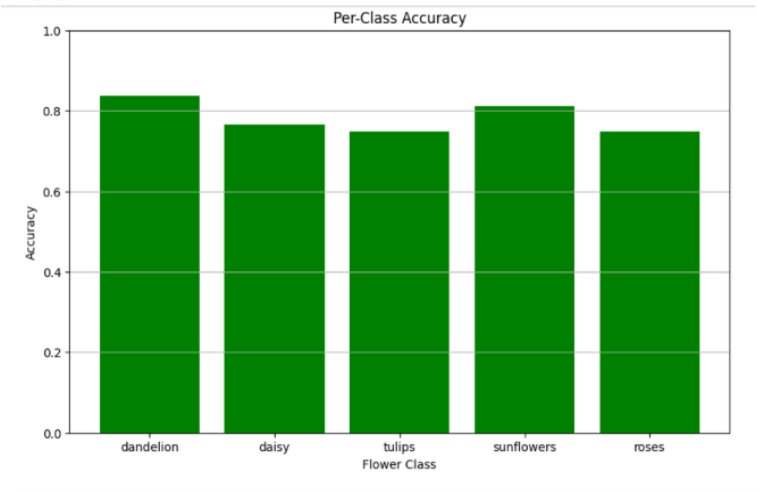
**Performance Metrics Used**

- **Accuracy**: Measures the overall correct predictions out of total predictions.
- **Validation Accuracy & Loss**: Helps monitor generalization to unseen data and detect overfitting.
- **Per-Class Accuracy**: Reveals which specific flower classes the model predicts well vs. poorly.
- **Batch-wise Accuracy**: Tracks consistency of performance across different data batches.

# Model Evaluation and Analysis

```
Evaluating model on dataset
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 940ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 985ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 896ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 979ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 822ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 903ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 944ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 2s 2s/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 2s 2s/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 2s 2s/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step
```



# Project Summary and Outcomes

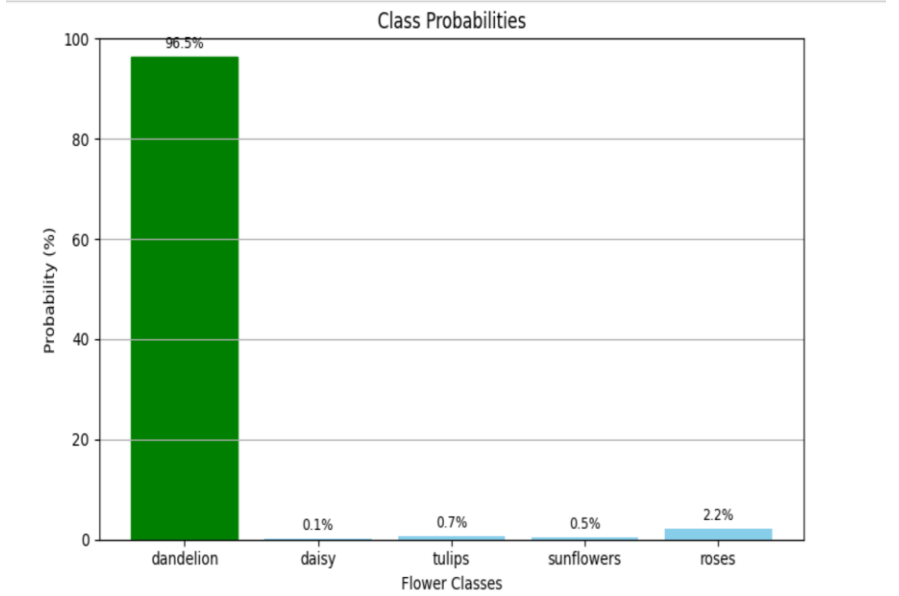
**Project Summary:**

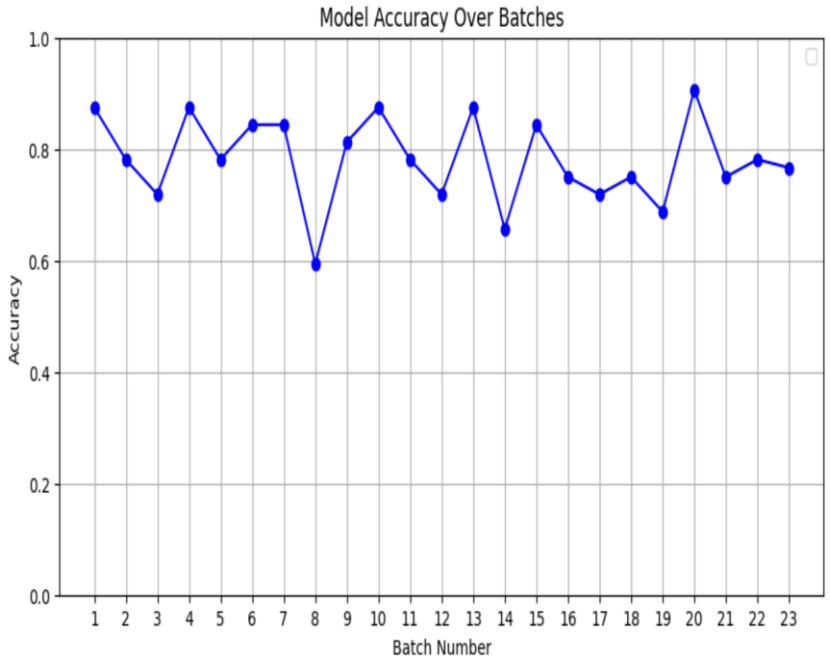
1. Built using a working  flower classifier using transfer learning.
2. Used MobileNetV2 + custom classification head
3. Achieved promising validation accuracy with fine-tuning
4. Demonstrated implementation of custom optimizer
5. Validated transfer learning’s effectiveness for image classification tasks

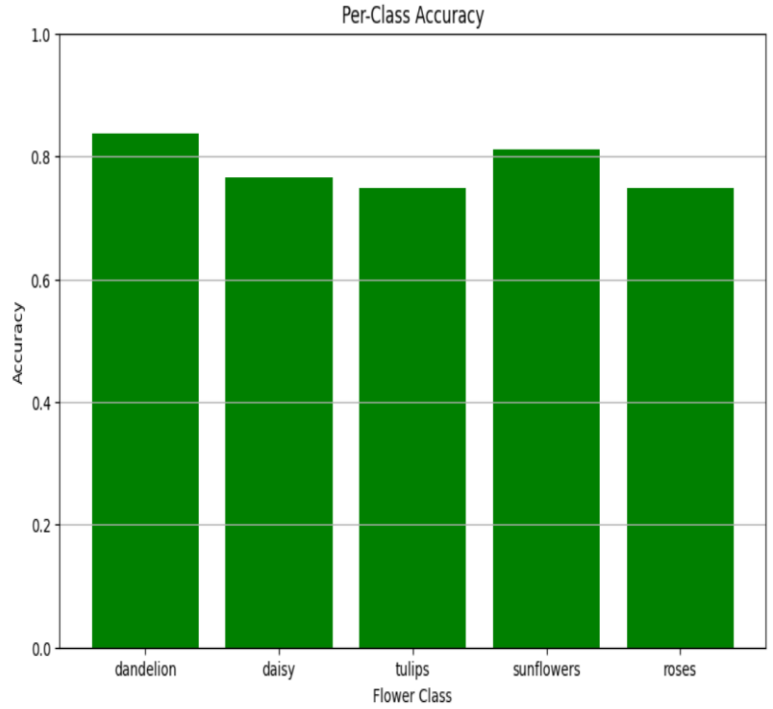
# Outcome:

```
Please enter the image path:
 dandelion.jpg
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 146ms/step

Predicted Flower: dandelion (96.47%)
```

# Future Improvements and Extension

1. **Usage of advanced Optimizers :** AdamW,Ranger, etc.
2. **Addition of Image Augmentation** for flipping,rotating and zooming.
3. **Test with different models :** EfficientNet,ViT
4. **Deployment:** Deploy as web or mobile app using Tensorflow.js
5. **Model Explainability:**Using Grad-CAM

# Reflections and Learning Outcomes

- **Understood Deep Learning** basics and model training workflow.
- **Explored Transfer Learning** using pre-trained MobileNetV2.
- Learned to **fine-tune layers** for better accuracy.
- Gained hands-on with **custom loss functions** (Focal Loss).
- Practiced **data augmentation** and normalization techniques.
- Built skills in using **TensorFlow Datasets (TFDS)**.
- Implemented **image preprocessing** and resizing.
- Visualized model performance through graphs and charts.
- Improved decision-making through **metric-based evaluation**.

# Supporting Materials:

**GitHub Link :**
A detailed project report, including objectives, methodology, evaluation, and learning outcomes, has also been prepared and can be accessed in the repository.

https://github.com/SanchitaH26/TransferLearning-for-Flower-Classification-using-TensorFlow-/blob/main/TransferLearning_project%20(1).ipynb

**Additional Resources**:
The complete source code, along with implementation details, data preprocessing pipeline, model architecture, training strategy, and evaluation scripts, is available on the GitHub repository:

https://github.com/SanchitaH26/TransferLearning-for-Flower-Classification-using-TensorFlow-/blob/main/Report.pdf