Module 1 -Project Report: Image Classification Using ResNet50

# Muhammad Usman Shahid

# F2022376068

# Workflow

* 1. Data Preparation  
   - Load training and testing image folders.  
   - Apply augmentation and normalization using torchvision.transforms.  
   - Split training data into training and validation sets.
* 2. Model Design  
   - Use a pre-trained ResNet50 model.  
   - Freeze most layers initially; unfreeze more layers later (layer2, layer3, layer4).  
   - Replace the final fully connected (FC) layer with a custom classifier.
* 3. Training Strategy  
   - Use Mixup data augmentation for better generalization.  
   - Train for 30 epochs using CrossEntropyLoss with label smoothing.  
   - Use AdamW optimizer and ReduceLROnPlateau scheduler.  
   - Implement gradient clipping and automatic mixed precision (AMP).
* 4. Evaluation  
   - Predict on validation and test sets.  
   - Evaluate using metrics: Accuracy, Precision, Recall, F1 Score, Top-1 and Top-3 Accuracy.  
   - Visualize the confusion matrix and loss/accuracy curves.
* 5. Submission  
   - Save test predictions in a CSV file for submission.

# Abstract

In this project, a robust deep learning pipeline was implemented for multi-class image classification involving 16 categories. Leveraging the pre-trained ResNet50 architecture with selective fine-tuning, the model was trained on a diverse dataset using advanced augmentation techniques including Mixup. Key metrics such as top-1/top-3 accuracy, precision, recall, and F1 score were used for model evaluation. The system achieved strong performance on the validation set, confirming the effectiveness of transfer learning and data regularization strategies. This pipeline is scalable and applicable to a wide range of visual classification tasks.

# Methodology

## 1. Data Augmentation & Preprocessing

Training Transforms: Random crops, flips, color jittering, grayscale, erasing, normalization.  
Validation/Test Transforms: Resize + normalization.

## 2. Model Architecture

Base Model: ResNet50 with pre-trained weights (ResNet50\_Weights.DEFAULT).  
Custom Classifier: Linear -> BatchNorm -> ReLU -> Dropout -> Linear(num\_classes)

## 3. Training Settings

Loss Function: CrossEntropy with label smoothing (0.2).  
Optimizer: AdamW (lr=0.0005, weight decay 1e-2).  
Scheduler: Reduce LR on plateau (monitors validation accuracy).  
Mixup Augmentation: Improves robustness by blending input images and labels.  
Mixed Precision Training: Improves performance using autocast() and GradScaler().

## 4. Validation & Metrics

Computed on validation split (20% of the dataset).  
Used metrics:  
 - Top-1 Accuracy  
 - Top-3 Accuracy  
 - Precision, Recall, F1-score (macro average)  
Visualizations: Loss and accuracy curves, confusion matrix.

## 5. Test Prediction

Custom TestDataset used to load and predict unseen images.  
Predictions mapped to class labels and saved to submission.csv.

# Results

## Evaluation Metrics on Validation Set:

|  |  |
| --- | --- |
| Metric | Score |
| Top-1 Accuracy | 91.18% |
| Top-3 Accuracy | 98.37% |
| Precision | 94.54% |
| Recall | 86.55% |
| F1 Score | 90.06% |

Visualization Summary:  
- Training vs Validation Loss: Training loss gradually reduced; validation loss plateaued after ~20 epochs.  
- Accuracy Curve: Consistent improvement over epochs; slight overfitting mitigated with scheduler and Mixup.  
- Confusion Matrix: Minor class confusion, but overall classification was accurate and balanced.

# Conclusion

This project successfully implemented a high-performing deep learning classification system using ResNet50 with a strong training pipeline incorporating Mixup augmentation and progressive unfreezing. The use of label smoothing and AMP contributed to stable and efficient training. Achieving over 91% top-1 accuracy and 98% top-3 accuracy demonstrates the effectiveness of transfer learning and regularization in deep image classification tasks. This approach can be extended to other classification challenges with minimal changes.

## Training Curves

The following plot shows the loss and accuracy curves over 30 training epochs. It helps visualize how the model performance evolved during training and validation. The dashed lines represent the final test loss and test accuracy values.

