A DEEP LEARNING PROJECT

FROM SCRATCH TO SMART

SERGIO EGUAKUN
CHAITANYA DIWAKAR
KATHERINE TORIAN

OVERVIEW

GOALS & OBJECTIVES

DATASET SETUP

CNN MODEL

TRANSFER LEARNING

KEY TAKEAWAYS

GOALS & OBJECTIVES

BUILD A BASELINE CNN FOR IMAGE CLASSIFICATION

Establish a simple benchmark model to understand CIFAR-10 and measure improvements against it.

EXPLORE STRATEGIES TO IMPROVE CNN PERFORMANCE

Apply techniques like data augmentation, batch normalization, dropout, and L2 regularization to enhance feature learning and reduce overfitting.

LEVERAGE TRANSFER LEARNING WITH PRETRAINED MODELS

Adapt state-of-the-art architectures (MobileNetV2, EfficientNetB0) to CIFAR-10 and compare their effectiveness against custom CNNs.

EVALUATE AND COMPARE MODELS TO IDENTIFY THE BEST APPROACH

Systematically assess test accuracy across models to highlight the impact of architectural choices and demonstrate how transfer learning achieves top performance (EfficientNetB0: ~93%).

DATASET SETUP

OUR DATASET

- 60,000 color images (32×32, RGB) across 10 classes (airplane, car, bird, cat, deer, dog, frog, horse, ship, truck).
- 50,000 training images + 10,000 test images.

DATA SPLITS & LABELS

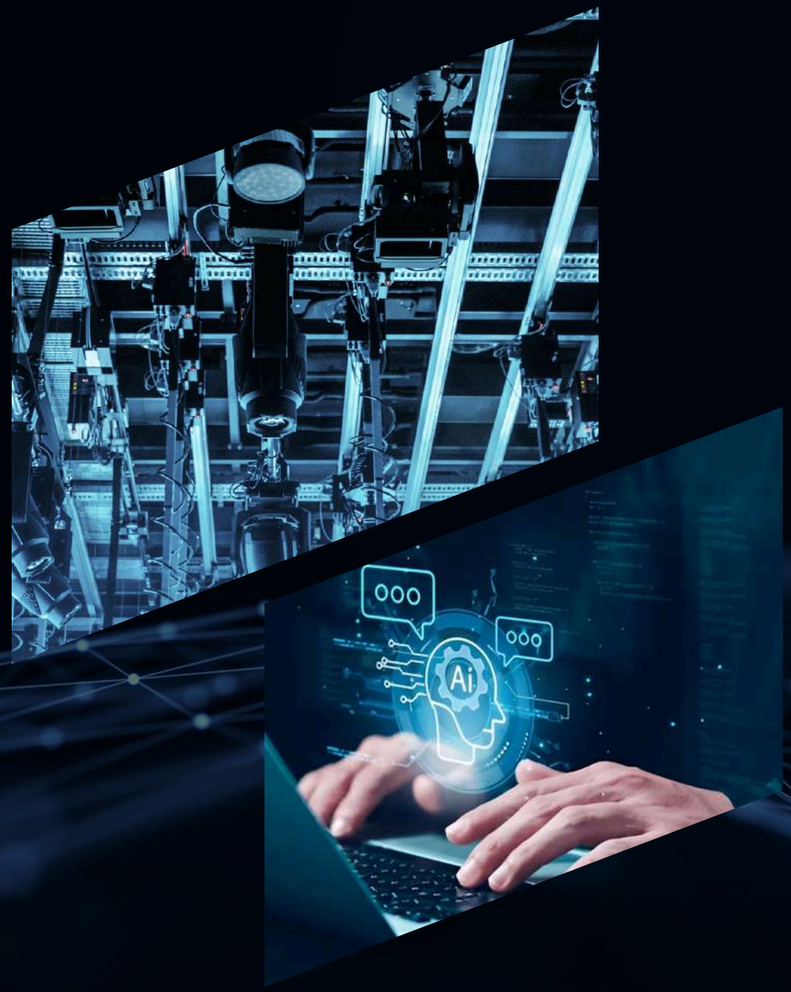
- Used cifar10.load_data() for train/test split.
- CNN models: integer labels + sparse_categorical_crossentropy.
- Transfer learning models: one-hot labels + categorical_crossentropy.

PREPROCESSING

Pixel values scaled to [0,1] (default) or preprocessed via model-specific functions (e.g., preprocess_input for MobileNetV2, EfficientNet).

EVALUATION METRIC:

Accuracy (training, validation, and test).



CNN MODELS

1- BASELINE CNN

- Architecture: Conv32 → MaxPool → Conv64 → MaxPool → Flatten → Dense128 → Dense10
- Features: Simple, interpretable, uses sparse categorical crossentropy
- Test Accuracy: 67.7%
- Takeaway: Overfits easily, limited feature extraction

2- BASELINE CNN + EARLY STOPPING + DATA AUGMENTATION

- Techniques: EarlyStopping (patience=3), Data Augmentation (flip, shift, zoom)
- Test Accuracy: 69.5% (ES), 74.2% (ES + Augmentation)
- Takeaway: Augmentation significantly improves generalization

3- STRONGER CUSTOM CNN (MODEL 2)

- Architecture: 3 blocks: Conv → BatchNorm → Conv → BatchNorm → MaxPool → Dropout
- Filters: 32 → 64 → 128, Dense256 before classifier
- Regularization: L2 + Dropout
- Test Accuracy: 84.3%
- Takeaway: Deeper network + BatchNorm + regularization reduces overfitting and captures richer features

TRANSFER LEARNING

MOBILENETV2

- Input resized: 32×32 → 96×96
- Pretrained on ImageNet, base frozen
- Custom head: GAP → Dropout(0.3) → Dense10
- Test Accuracy: 85.6%
- Key point: Lightweight model, pretrained features generalize well

MOBILENETV2

- Input resized: 32×32 → 224×224
- Labels one-hot, base pretrained on ImageNet
- Training:
  - Phase 1: freeze base, train head (10 epochs, LR=1e-4)
  - Phase 2: fine-tune last ~30 layers (10 epochs, LR=1e-5)
- Custom head: GAP → Dropout0.3 → Dense10
- Test Accuracy: 92.7%
- Key point: Fine-tuning pretrained features gives best generalization

KEY TAKEAWAYS

EARLY STOPPING

Prevents overfitting, small improvement

DATA AUGMENTATION

Improves generalization significantly (+5 points).

BATCHNORM + L2 + DROPOUT

Deeper, regularized CNN reduces overfitting and captures richer features (+10 points).

MOBILENETV2

Pretrained features outperform custom CNNs (+0.6 points vs Model 2), lightweight and efficient.

EFFICIENTNETB0

Fine-tuned transfer learning achieves state-of-the-art accuracy (92–93%), best generalization.

ABOUT THE FUTURE...

- Experiment with larger or more advanced architectures (e.g., EfficientNetB3/B4) for potential accuracy gains.
- Explore semi-supervised or self-supervised learning to reduce dependency on labeled data.
- Apply transfer learning strategies to other small or domain-specific datasets.
- Combine multiple models (ensembles) to push accuracy even higher.
- Investigate optimization techniques and learning rate schedules for more efficient training.

THANK YOU

ANY QUESTIONS?