Experimental Analysis of ANN, CNN, and Hybrid CNN+ANN on CIFAR-10

Ayan Asif
Section CS-7A, Roll No: 22I-1097

Abstract

This report presents a systematic exploration of neural network architectures trained on the CIFAR-10 dataset. Three models were implemented from scratch using PyTorch: a fully-connected feedforward Artificial Neural Network (ANN), a Convolutional Neural Network (CNN), and a hybrid CNN+ANN model. Each model was trained and evaluated under identical experimental conditions to assess performance differences in representational capacity, inductive bias, and feature extraction efficiency. Through controlled experiments varying activation functions, normalization, weight decay, and learning rate scheduling, we analyze the effect of architectural and optimization choices on model accuracy and generalization. The CNN achieved the highest test accuracy of 75.4%, the hybrid model reached 71.4%, while the ANN baseline achieved 56.9%. Our findings highlight the fundamental advantage of convolutional feature extraction in visual recognition tasks and demonstrate how thoughtful design in simple architectures can yield competitive results without transfer learning or pretrained weights.

Keywords: CIFAR-10, Feedforward ANN, Convolutional Neural Network, Deep Learning, PyTorch, Regularization, Optimization.

1 Introduction

The purpose of this assignment was to experimentally investigate how far fully-connected and convolutional architectures can perform on an image classification task without relying on transfer learning or prebuilt models. CIFAR-10 consists of 60,000 color images (32×32 pixels) across ten categories (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck). The challenge lies in training simple, self-designed networks that extract discriminative features under constrained conditions.

This report covers three progressively stronger architectures:

- A fully-connected **ANN** (dense-only model),
- A handcrafted **CNN** using convolution, pooling, dropout, and normalization,
- A **Hybrid CNN+ANN** combining convolutional feature extraction with dense classification layers.

Each model was trained on identical training and test splits using stochastic optimization. We analyze how representational power and inductive bias affect classification performance.

2 Dataset and Preprocessing

The CIFAR-10 dataset was loaded directly from `torchvision.datasets.CIFAR10`. Images were normalized to zero mean and unit variance using per-channel statistics:

$$x' = \frac{x - \mu}{\sigma}$$

where $\mu = (0.4914, 0.4822, 0.4465)$ and $\sigma = (0.2023, 0.1994, 0.2010)$.

For CNN and hybrid models, light data augmentation was applied:

- Random horizontal flips
- Random crops (padding = 4)

All models were trained with a batch size of 128. Since training was CPU-based, runs were limited to 5–10 epochs each.

3 Model Architectures

3.1 Fully Connected ANN

The ANN comprised three hidden layers with batch normalization and dropout:

$$3072 \rightarrow 1024 \rightarrow 512 \rightarrow 256 \rightarrow 10$$

Each hidden layer used ReLU activation, 0.3 dropout, and BatchNorm1d. Training used the Adam optimizer (lr=1e-3) and CrossEntropy loss.

3.2 Convolutional Neural Network (CNN)

The CNN contained three convolutional blocks:

$$3 \rightarrow 32 \rightarrow 64 \rightarrow 128$$

Each block used a 3×3 convolution, ReLU, and MaxPooling(2,2). After flattening, two dense layers (2048→256→10) completed the classifier. Dropout (0.5) was applied for regularization. BatchNorm and a CosineAnnealingLR scheduler stabilized training.

3.3 Hybrid CNN+ANN

The hybrid model reused the CNN's feature extractor (three convolutional blocks) and attached a deeper ANN classifier:

$$128 \times 4 \times 4 \rightarrow 512 \rightarrow 256 \rightarrow 10$$

Both CNN and ANN parts were trained jointly, with weight decay (1e-4) for regularization.

4 Experimental Setup

All experiments used:

- Framework: PyTorch
- Optimizer: Adam
- Loss: CrossEntropy
- Scheduler: CosineAnnealingLR (for CNN)
- Random seed fixed for reproducibility

We performed small ablation studies:

1. **Activation Function Test:** ReLU vs. LeakyReLU in ANN.
2. **Weight Decay:** L2 regularization in ANN.
3. **Normalization:** BatchNorm in CNN.
4. **Learning Rate Schedule:** StepLR and CosineAnnealing.

5 Results

Table 1: Model Performance Comparison on CIFAR-10

Model	**Epochs**	**Train Acc (%)**	**Test Acc (%)**	**F**
Baseline ANN (Sigmoid)	10	60.46	56.95	Weak sp
ANN (ReLU)	5	62.99	58.0	Improve
ANN (ReLU + Weight Decay)	5	61.01	59.0	Slightly be
CNN (Baseline)	5	75.17	75.19	Excelle
CNN + BatchNorm	5	73.45	74.69	Sta
CNN + CosineAnnealingLR	5	77.34	75.43	Highest
Hybrid CNN+ANN	5	68.23	71.40	Strong trade-off betw

5.1 Analysis

The ANN's low accuracy demonstrates its limited ability to model spatial hierarchies in pixel data. ReLU and regularization slightly improve training stability. CNN's superior performance arises from weight sharing and local receptive fields, enabling translation invariance and better feature abstraction. The hybrid model benefits from CNN feature extraction but incurs heavier computation due to deeper fully-connected layers.

6 Discussion

6.1 Representational Capacity

The CNN and hybrid models outperform the ANN due to convolution's ability to capture local dependencies and compositional features. The ANN must learn pixel-wise relationships from scratch, requiring more parameters and data to reach equivalent accuracy.

6.2 Regularization and Normalization

Batch normalization and dropout substantially improved convergence and reduced overfitting. Weight decay helped prevent exploding weights, though its effect was modest on small networks.

6.3 Learning Dynamics

The learning rate schedule experiment showed that aggressive step decay harms training, whereas cosine annealing provides smoother convergence. The hybrid's slightly lower accuracy than CNN stems from its heavier classifier, which may overfit small batches.

7 Conclusion

This study demonstrates that handcrafted, non-pretrained neural networks can achieve over 70% accuracy on CIFAR-10 using systematic experimentation and careful tuning. Key lessons include:

- Convolutional feature extraction is essential for visual tasks.
- Simple normalization and scheduling improve training stability.
- Hybrid architectures provide a middle ground between CNN's feature hierarchy and ANN's flexible classification.

Future work could explore progressive resizing, data augmentation strategies like CutMix or MixUp, and ensembles of lightweight models.

8 References

1. Krizhevsky, A. (2009). *Learning Multiple Layers of Features from Tiny Images.* University of Toronto.
2. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning.* MIT Press.
3. PyTorch Documentation: `https://pytorch.org/docs/stable/`