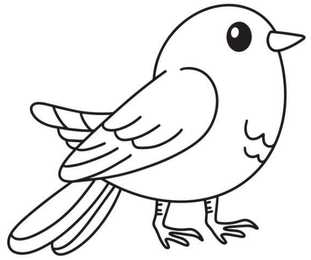
Neural Net-Works 👍🏾

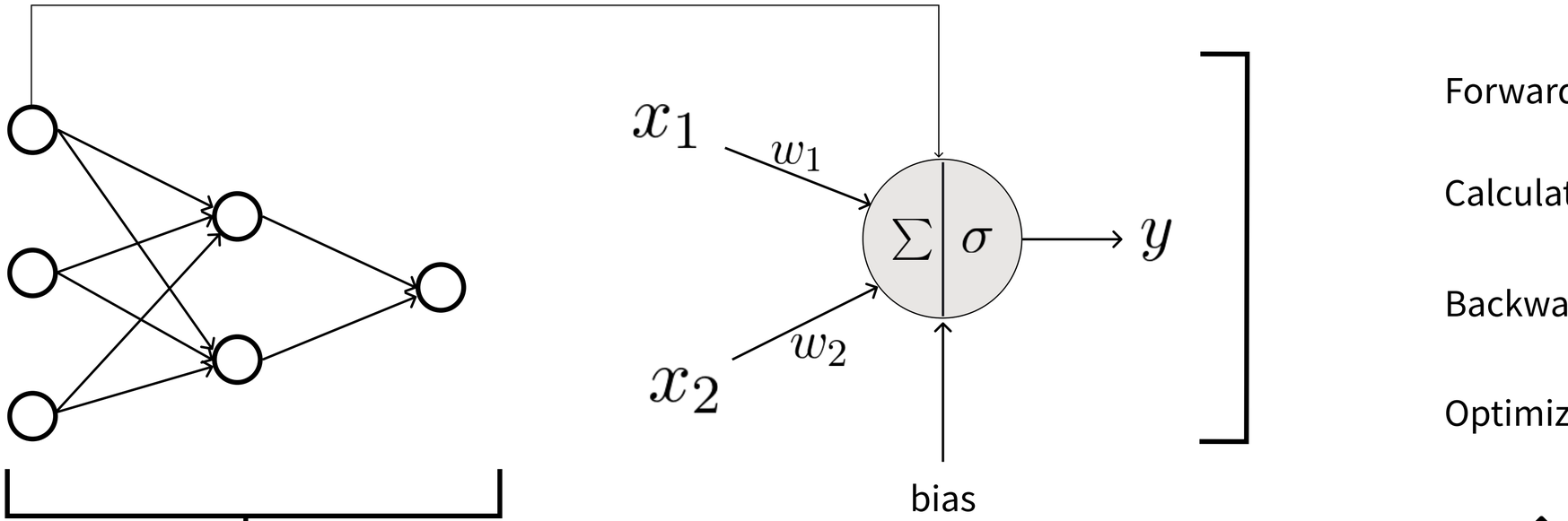
How do machines learn to see?

Can you code up a program to tell if this image is a bird or not?



Quite clever! How about you try for these...

Deep Feed Forward Networks



Forward pass $y = \sigma(\sigma(\sigma(\sum_{i=1}^{n} w_i x_i)))$

Calculate the loss $\mathcal{L}(y, \hat{y})$

Backward pass

Optimization

Stages of Learning

Preparing the Data

- Dataset: CINIC-10 (10 categories, RGB, 32 X 32)
- Split into Train / Validation / Test (balanced per class)
- Resizing: images → 32×32 for uniformity
- Flattening:

$$H \times W \times C = 32 \times 32 \times 3 = 3{,}072 \text{ features}$$

- Normalization: Standardize pixel values ([-1, 1])
- Data Augmentation: Improves generalization & over-fitting.

SECTION NAME

Feedforward DNN

Tuning the Learning Process

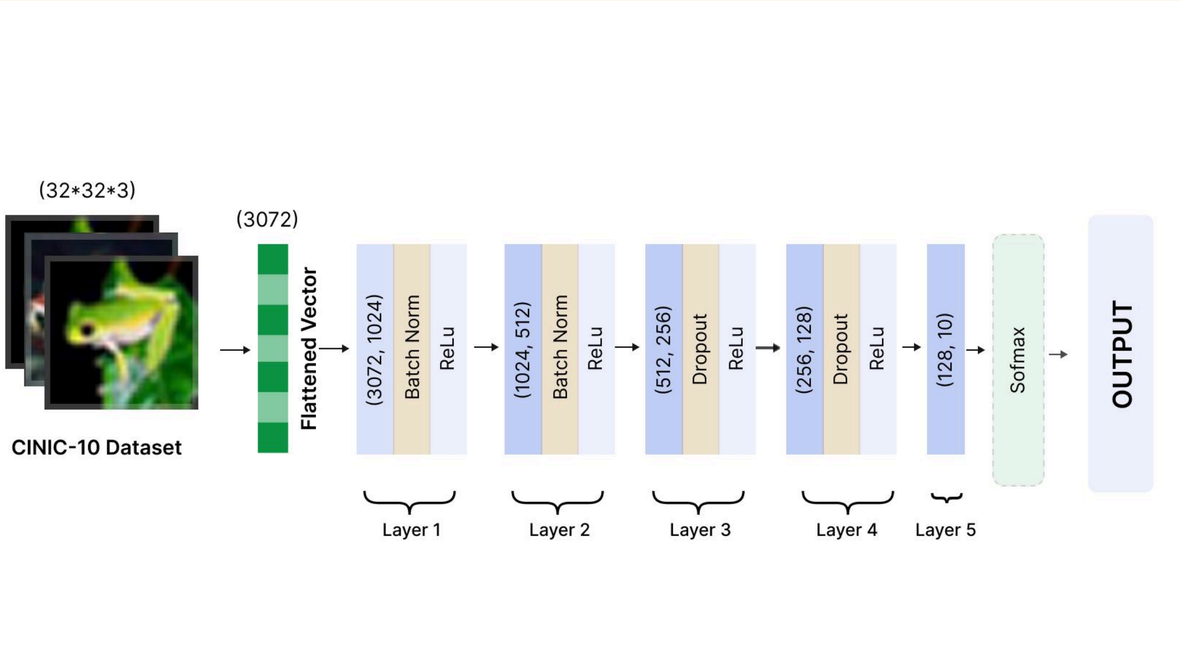
Grid Search :

- Systematically tested combinations of the learning rate, weight decal, batch sizes etc
- Selected best values based on validation accuracy

Hidden Layers : 3 - 6 hidden layers

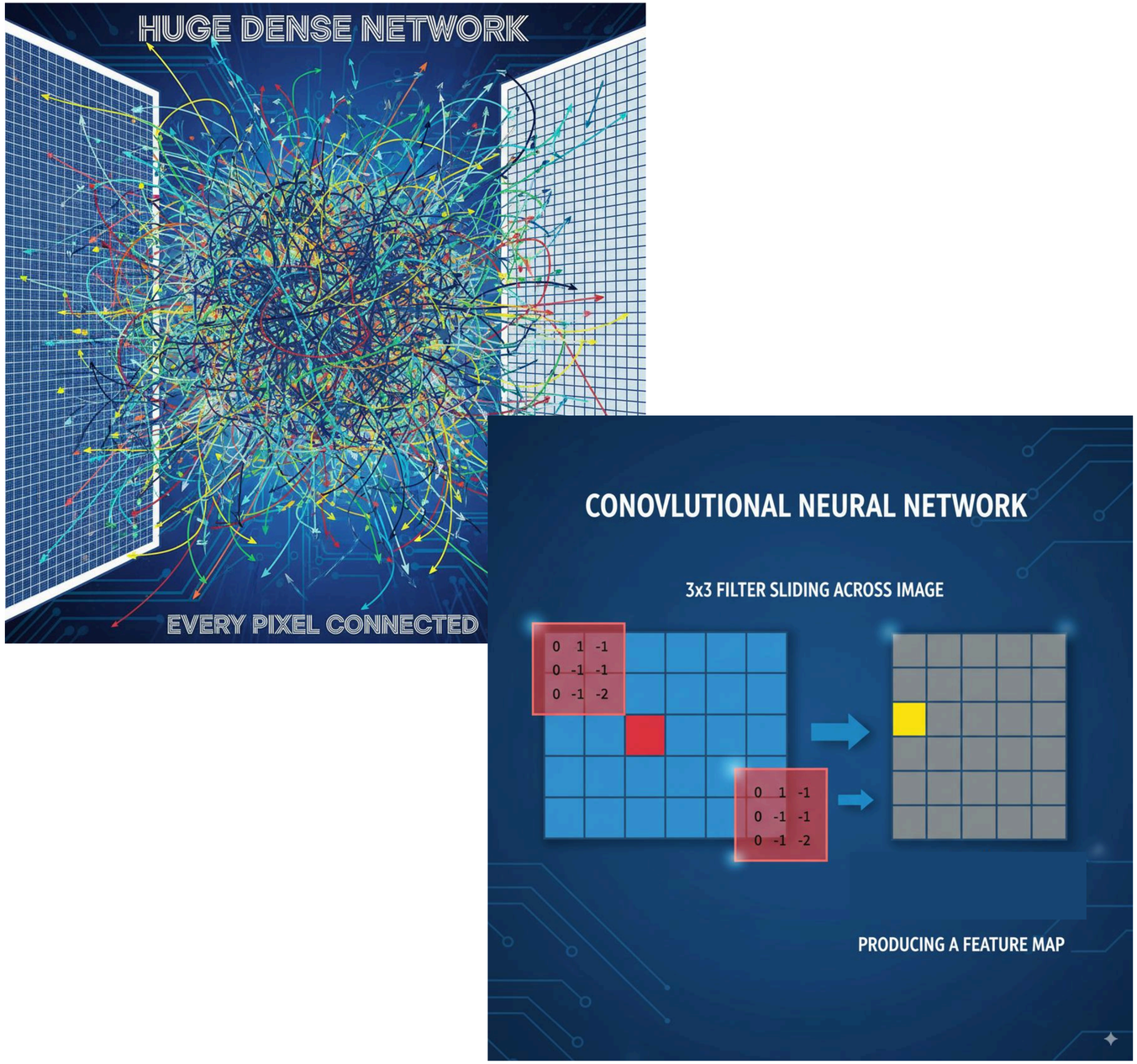
Optimizer: AdamW & Adam

Loss : CrossEntropyLoss

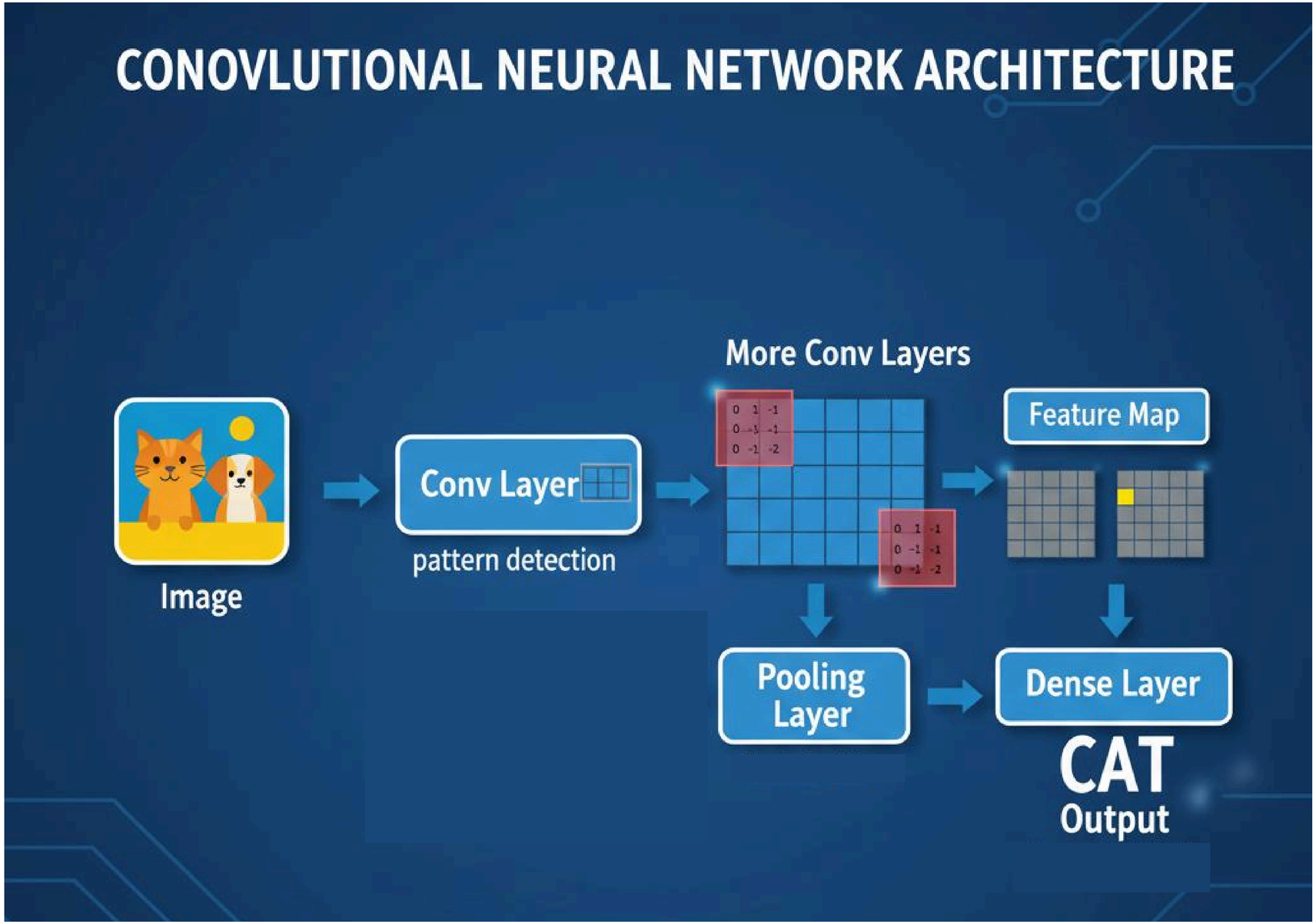


From DNNs to CNNs

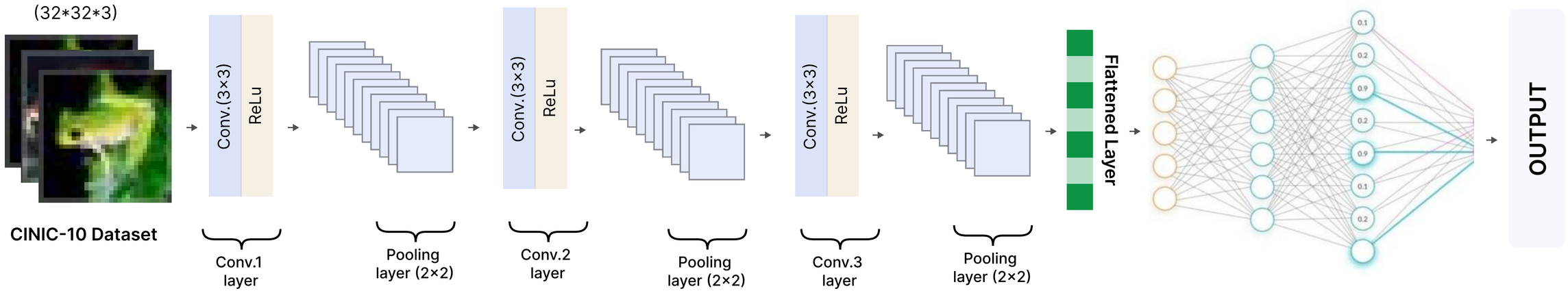
- DNNs struggle with images (millions of weights)
- CNNs use small filters (kernels) → fewer weights
- Filters learn to detect patterns (edges, textures)



Layers of a CNN

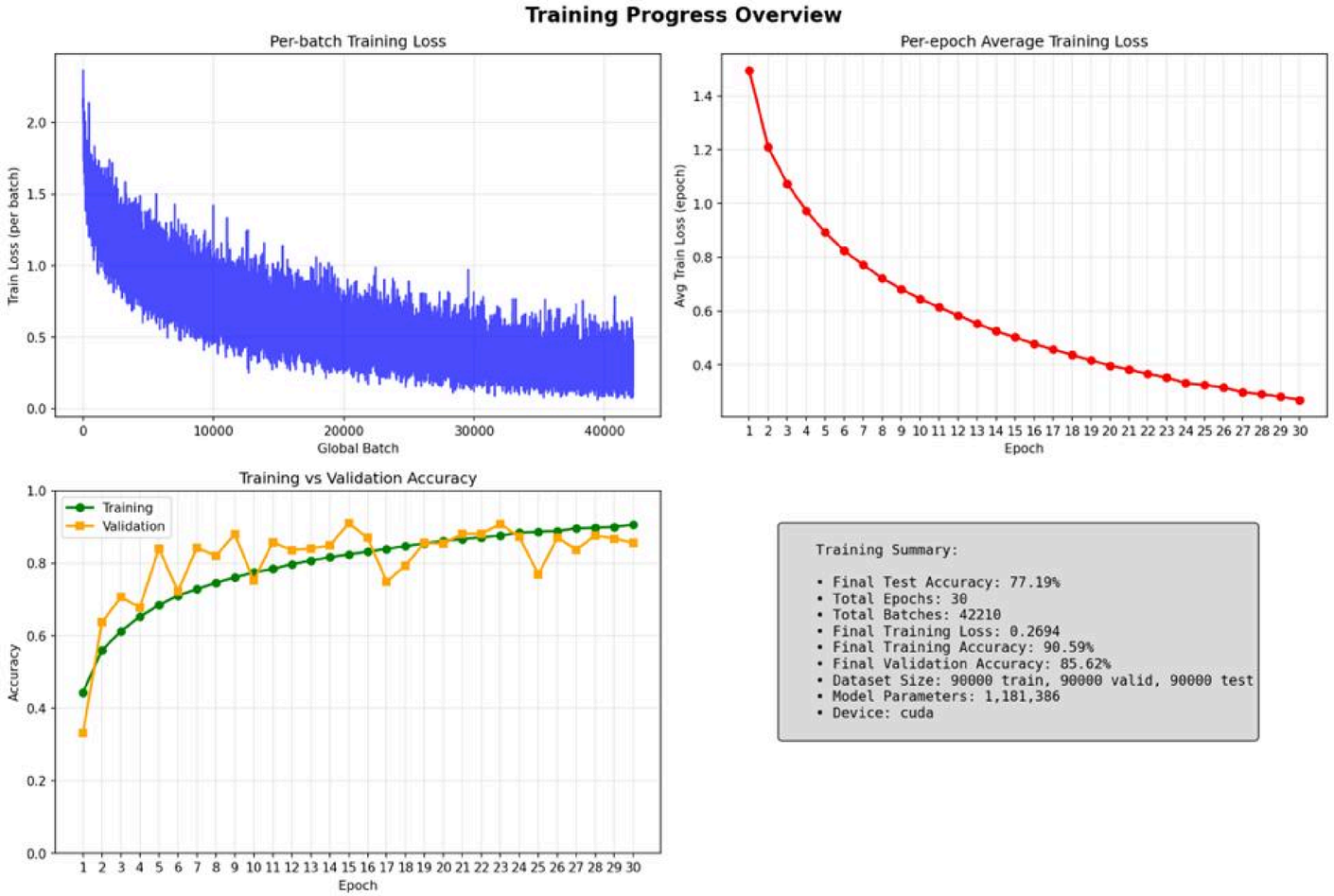


OUR CNN ARCHITECTURE



Versions:

Model	Key Features	# of Parameters
Simple CNN	3 conv layers, 3 FC layers	1,276,234 **(~ 1.3M)**
Advanced CNN	6 conv layers, BatchNorm, Global pooling	1,181,386 **(~1.18M)**



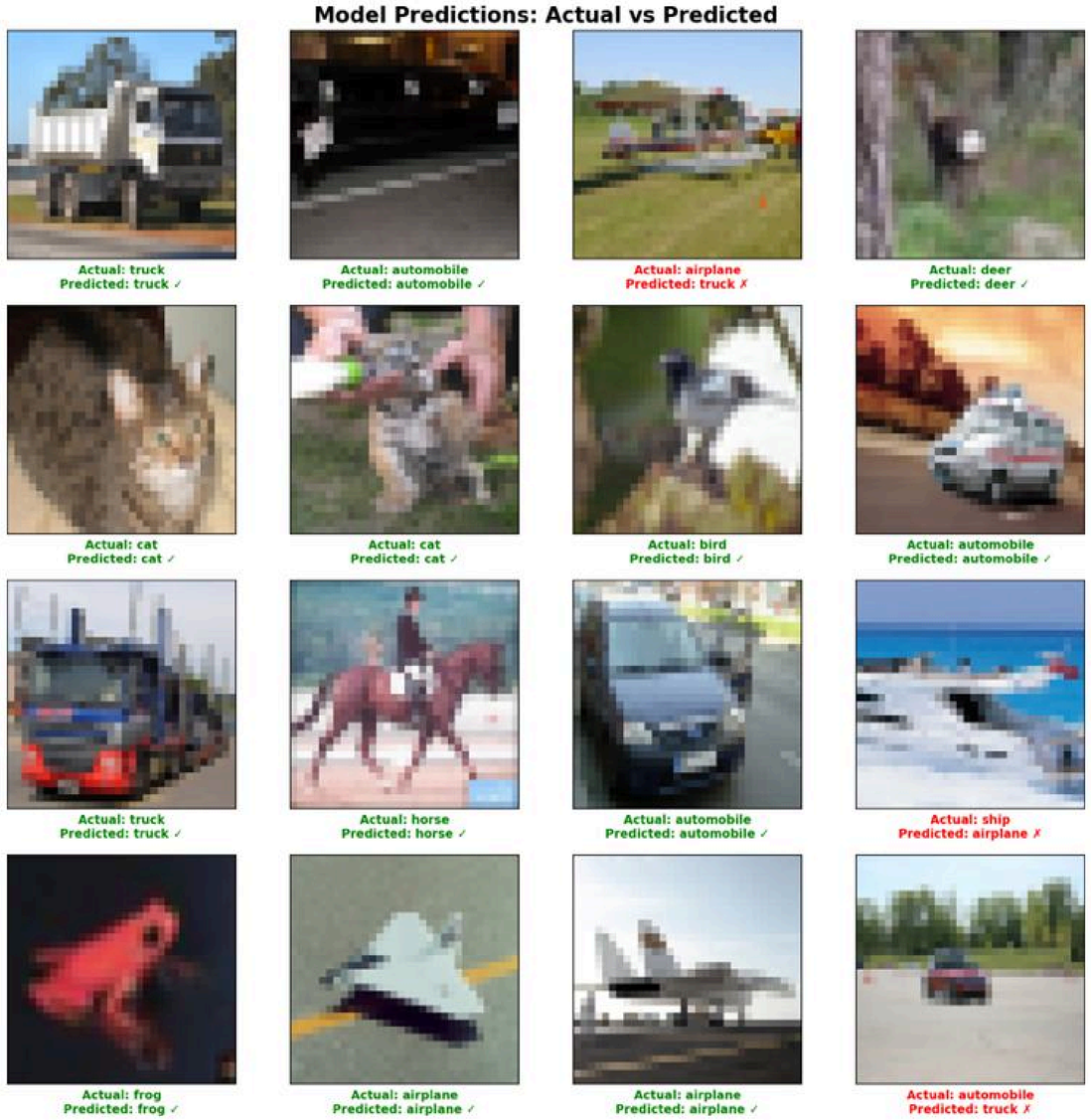
Inference:

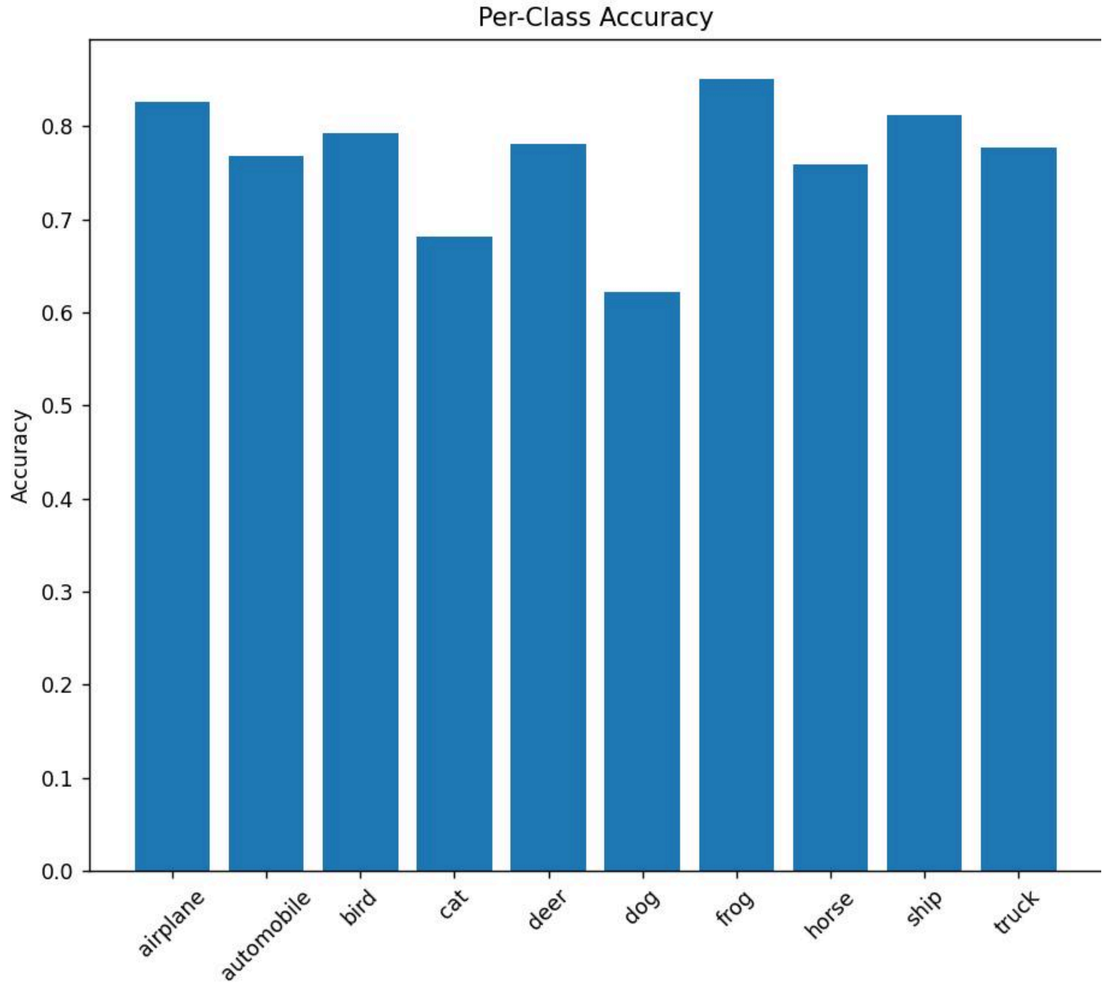
- Training accuracy: 40% → 90.59%
- Validation accuracy: 35% → 85.62%
- Loss reduction: 1.45 → 0.27 (81% reduction)

Overfitting Pattern:

- Training accuracy (90.59%) > Validation accuracy (85.62%). ~5% indicates mild overfitting.

OUR CNN ARCHITECTURE





Q. E. D.