Group 6

NEURAL NETWORKS IMAGE ANALYSIS

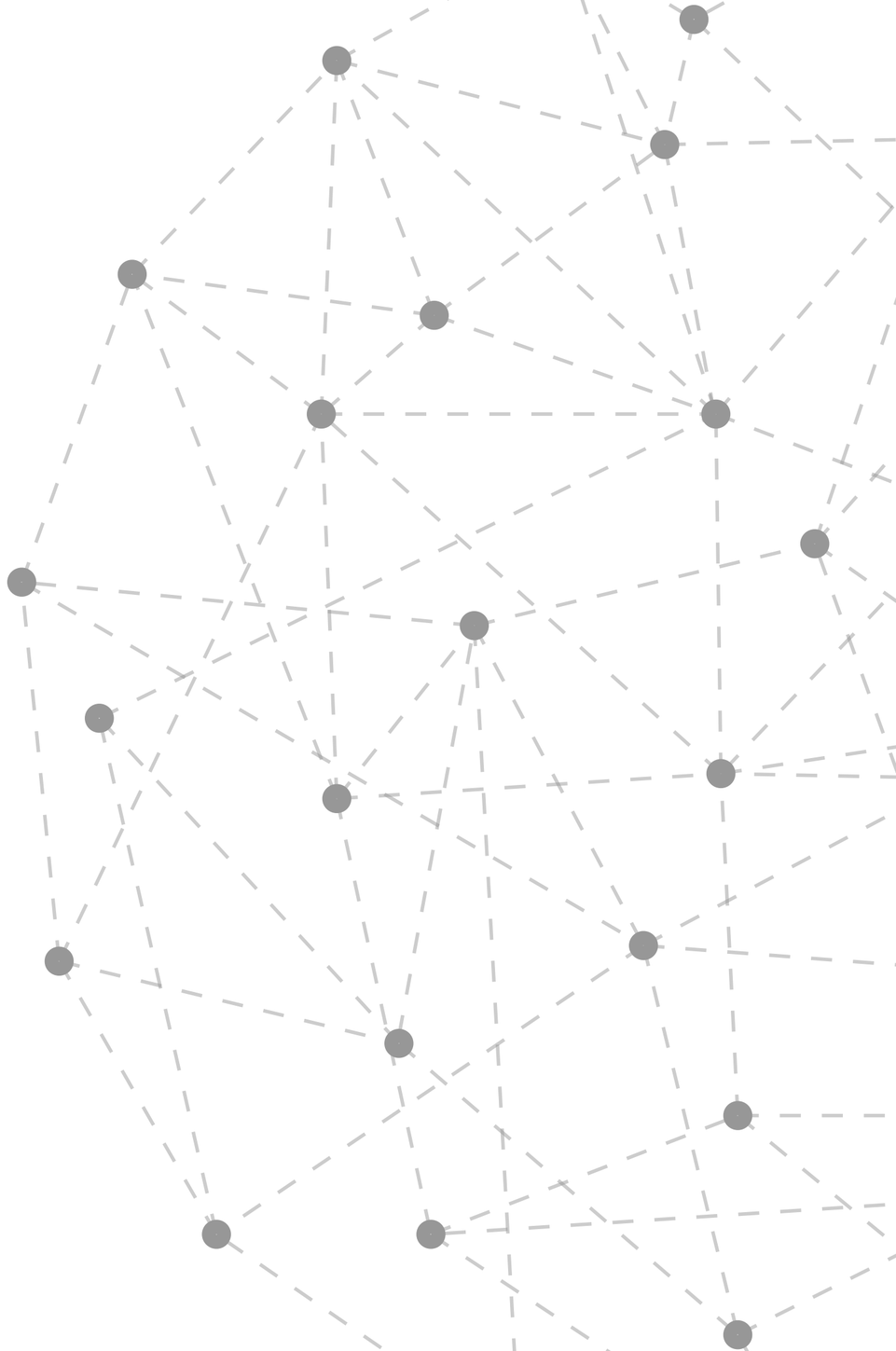


TABLE OF CONTENTS

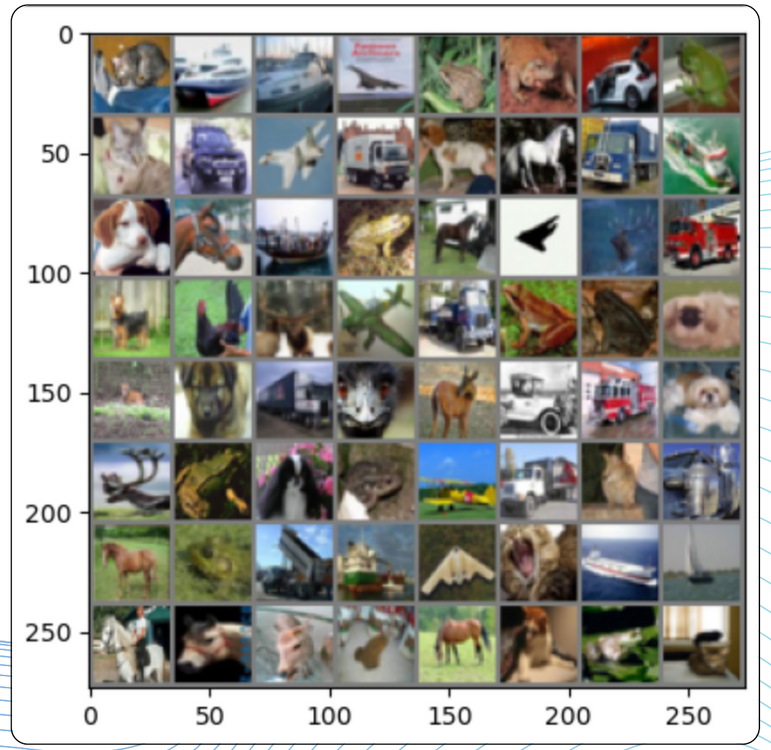
Presentation Agenda:

- Dataset Overview
- ANN Implementation
- CNN Implementation
- Model Evaluations
- Conclusion
- Questions

DATASET OVERVIEW

CIFAR-10 Dataset

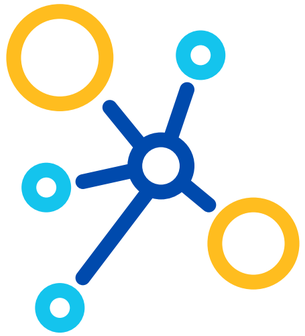
- Dataset containing **60,000** 32x32 colour images
- Images consist of the following 10 classes:
  - **'Plane', 'Car', 'Bird', 'Cat', 'Deer', 'Dog', 'Frog', 'Horse', 'Ship', 'Truck'**
- Each class consists of 6,000 images distributed across test & training sets
- **Training Set:** 5 batches of 10,000 images
- **Testing Set:** 1 batch of 10,000 images, 1000 per class




```
# Define transformations for the dataset
transform = transforms.Compose([
    transforms.ToTensor(),
    transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))  # Normalize the data
])

# Load CIFAR-10 dataset
trainset = torchvision.datasets.CIFAR10(root='./data', train=True, download=True, transform=transform)
testset = torchvision.datasets.CIFAR10(root='./data', train=False, download=True, transform=transform)

# Set up data loaders
train_loader = DataLoader(trainset, batch_size=64, shuffle=True)
test_loader = DataLoader(testset, batch_size=64, shuffle=False)
```



MODEL IMPLEMENTATIONS

ARTIFICIAL NEURAL NETWORK

Model Setup

- Define the neural network architecture, including 4 fully connected layers, ReLu activation functions, etc.
- Initialize the model and specify the loss function and optimizer.
  - Used the Stochastic Gradient Descent optimizer
  - Also tested with the Adam optimizer but yielded lower accuracy

Training

- Iterate through the dataset in batches.
- Perform forward pass, calculate loss, and backpropagate to update model parameters.

Evaluation

- Use the trained model to make predictions on the test set.
- Fine-tune the model by adjusting hyperparameters, architecture, and optimization strategies based on performance.

```
class ANN(nn.Module):
    def __init__(self):
        super(ANN, self).__init__()
        self.fc1 = nn.Linear(32 * 32 * 3, 512)  # Input size is 32x32x3, output size is 512
        self.fc2 = nn.Linear(512, 256) # Input size is 512 and output size is 256
        self.fc3 = nn.Linear(256, 128) # Input size is 256 and output size is 128
        self.fc4 = nn.Linear(128, 10)  # Input size is 256 and output size is 10 (number of classes in CIFAR-10)
```

CONVOLUTIONAL NEURAL NETWORK

Convolutional Neural Networks are commonly used for computer vision tasks. The convolution layer works by applying a filter (kernel) to an image, mathematically applying the dot product of the filter to a subsection of the image.

Model Setup

- Defined Convolutional Neural Network has 3 fully connected layers.
- Define loss function and optimizer then initialize.

Training

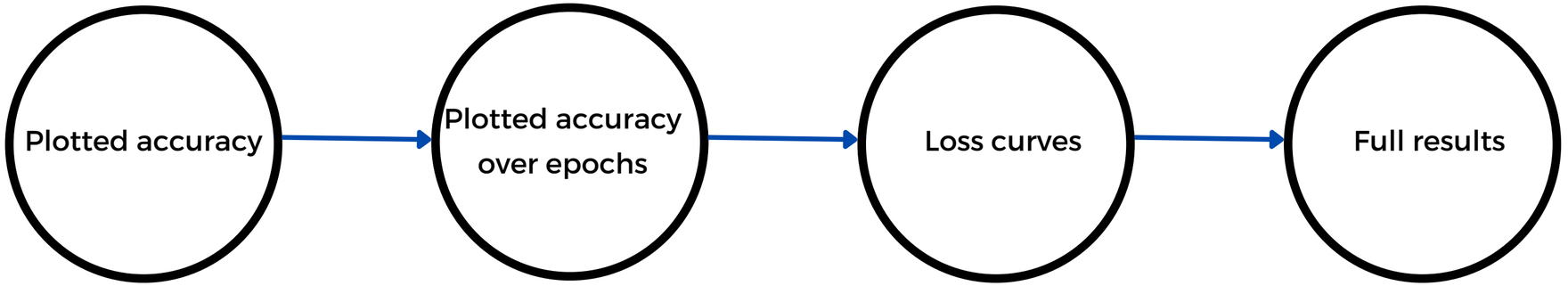
- Loop over the training set for a number of epochs.
- Forward propagate, calculate loss, backpropagate, update the weights, and track total loss.

Evaluation

- Use our trained model to classify test data.

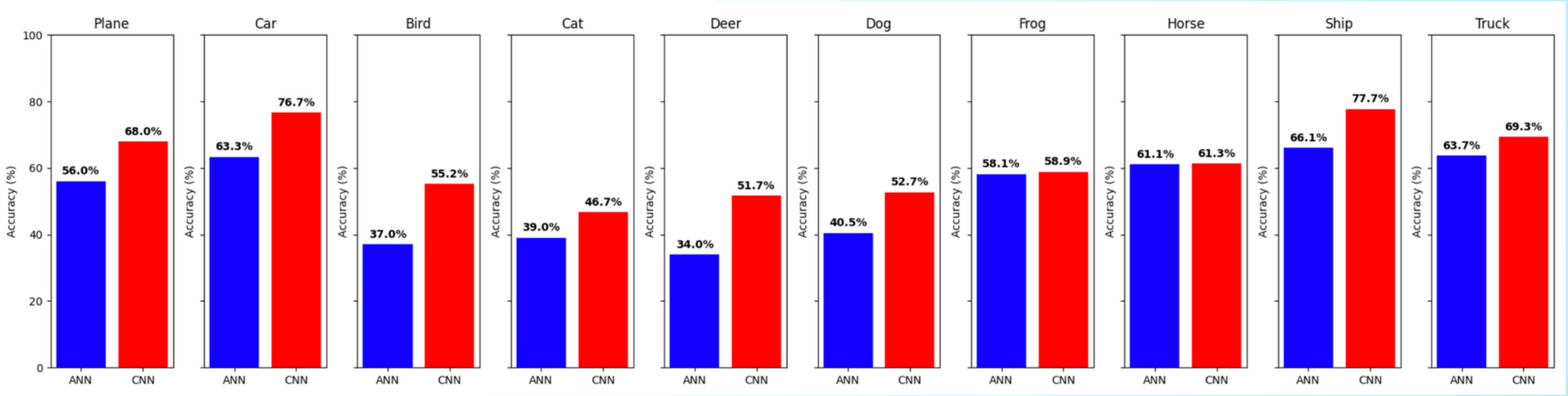
MODEL EVALUATIONS

- To compare both models, we will compare two different metrics
  - Loss calculations throughout the training
  - Accuracy in each class
    - Plotted as bar graph per class
    - Plotted per training iteration on each class
    - Plotted learning curves of accuracy over epochs
- CNN had a lower loss and better overall accuracy (61%) compared to ANN (51%)



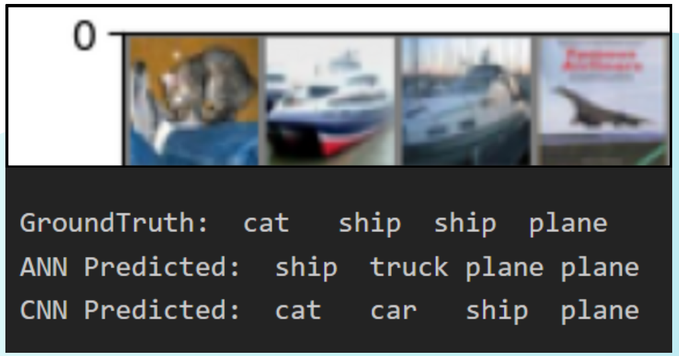
MODEL EVALUATIONS

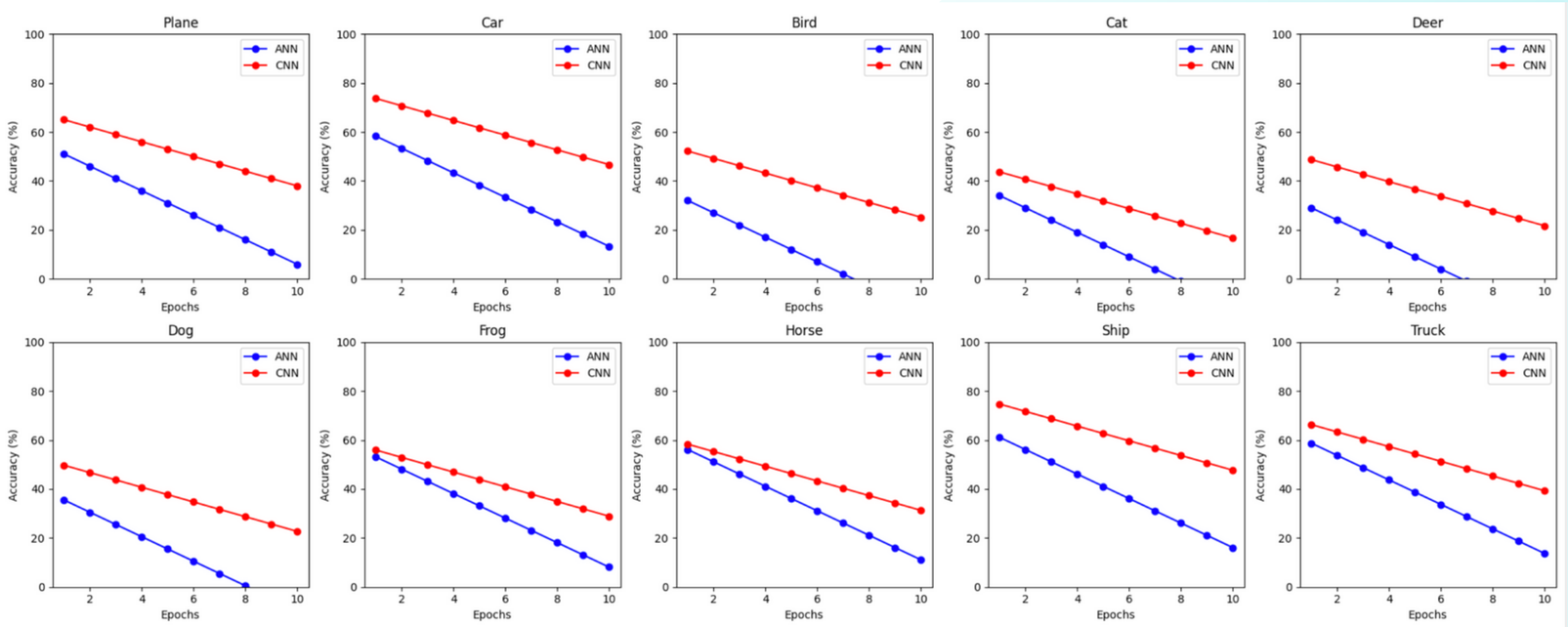
- Models testing performance metrics (**Accuracy of the model to each class**)



MODEL EVALUATIONS

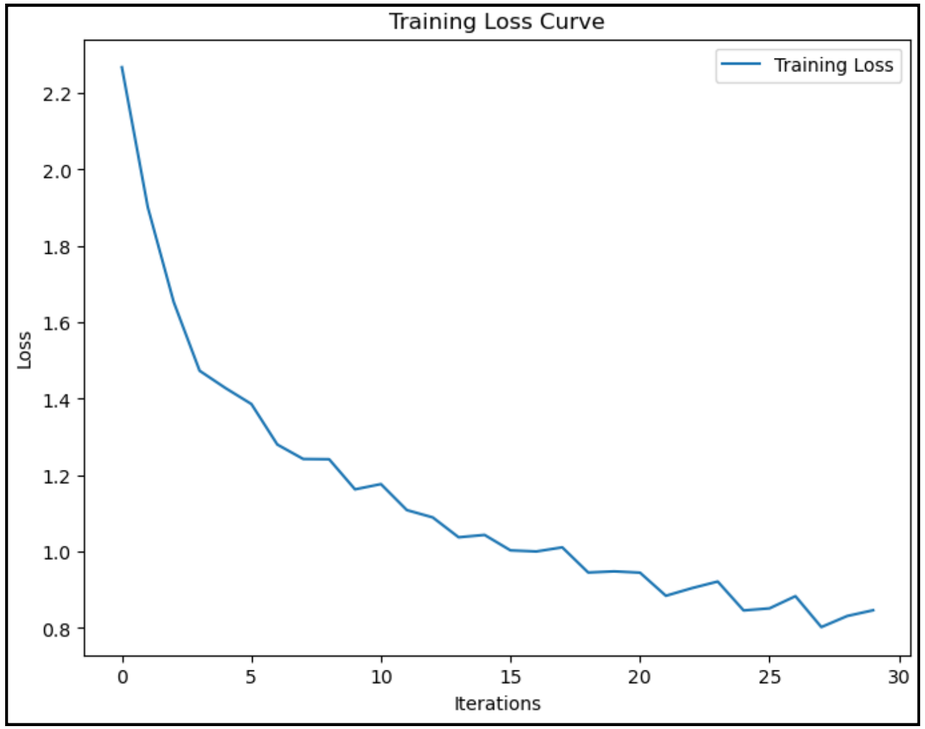
- The **learning curves for accuracy over epochs**, comparing models for different classes



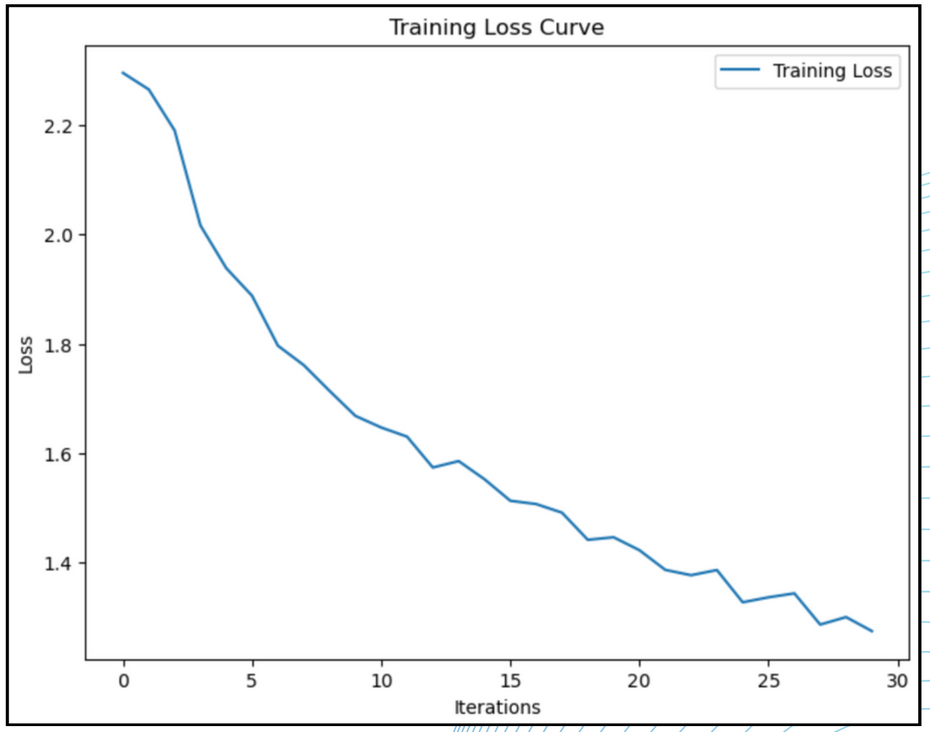


LOSS CURVES

CNN



ANN



RESULTS

CNN

Table 1: CNN Classification Performance

Class	Accuracy (%)
Plane	68.0
Car	76.7
Bird	55.2
Cat	46.7
Deer	51.7
Dog	52.7
Frog	58.9
Horse	61.3
Ship	77.7
Truck	69.3

ANN

Table 2: ANN Classification Performance

Class	Accuracy (%)
Plane	56.0
Car	63.3
Bird	37.0
Cat	39.0
Deer	34.0
Dog	40.5
Frog	58.1
Horse	61.1
Ship	66.1
Truck	63.7

CONCLUSION

KEY FINDINGS

- CNN performed 10% better overall on the CIFAR-10 dataset with an accuracy of 61% compared to ANN's accuracy of 51%.
- The potential for further exploration could be to use a different optimization function, preprocess the images differently, or change the architecture of the models.

1) Both models on average had higher accuracy with objects compared to animals

2) Both had similar loss curves in their training but CNN had a lower final loss by ~0.4

3) ANN performed the best with the 'Ship' class (66.1%) and the worst with the 'Deer' class (34.0%)

4) CNN performed the best with the 'Ship' class (77.7%) and the worst with the 'Cat' class (46.7%)

QUESTIONS?