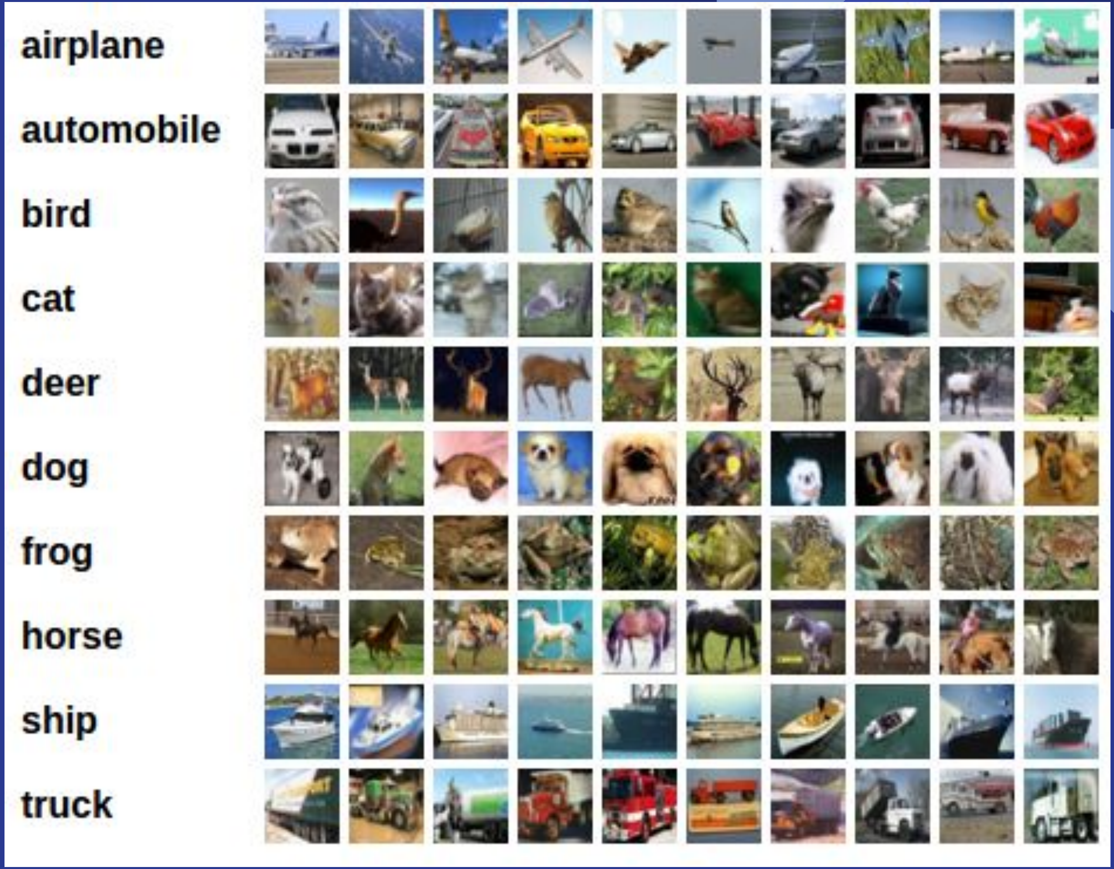
CIFAR-10 Image Classification

Cameron Wilson, Brandon Santore
CSCI-335
Rochester Institute of Technology
12/3/2025

Introduction



Problem

Given a dataset of CIFAR-10 images, our goal was to design a CNN to be as accurate as possible in classifying the images and compare it to traditional models.

Background

- We trained three different image-classification models
  - Convolutional Neural Network (CNN)
  - Random Forest
  - Linear Support Vector Machine (SVM)
- CNNs are generally considered to be the best for image classification, so this was our primary model.
- The traditional models were made for comparison

Hypothesis

Our hypothesis is that out of all of our models our CNN will perform the best as CNNs are commonly thought of as the best models for image classification.

Methods

CIFAR-10 Dataset

- We obtained the CIFAR-10 Dataset from the Tensorflow Keras datasets package
- The dataset contains 60,000 images
  - 50,000 training images
  - 10,000 testing images
- The images were 32x32 pixels and colored
- The dataset was divided into 10 classes
  - Airplane, Automobile, Bird, Cat, Deer, Dog, Frog, Horse, Ship, Truck
- The dataset is loaded into Numpy arrays, converted to floats, normalized, and the target arrays are flattened.

Traditional Models

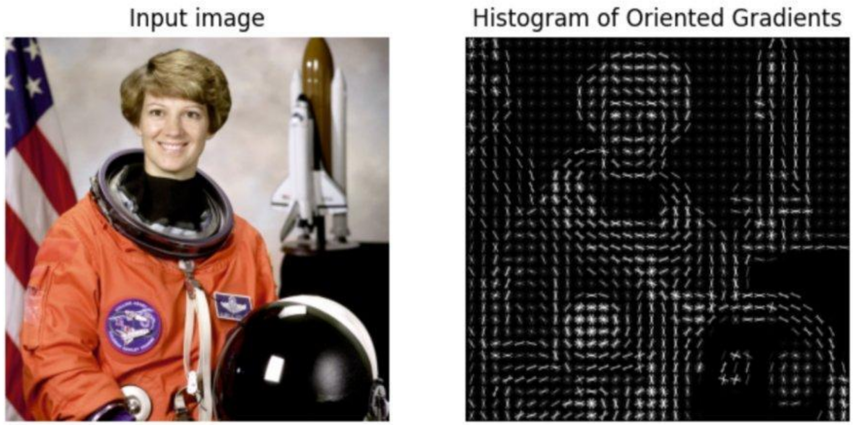
- We selected Random Forest and Support Vector Classifier as they are fairly good models for multi-classification
- Linear kernel was used for SVC, otherwise the large size of dataset would take too long to train

Preprocessing

- For the CNN:
  - Pixel values are normalized from [0, 255] to [0, 1]
  - Training data is split: 45,000 images for training and 5,000 for validation
  - One-hot encode the image labels into vectors to allow model to train off of them
- For the SVC and Random Forest:
  - Each image in the dataset is converted to greyscale
  - Features are extracted using Histogram of Oriented Gradients (HOG)

Histogram of Oriented Gradients

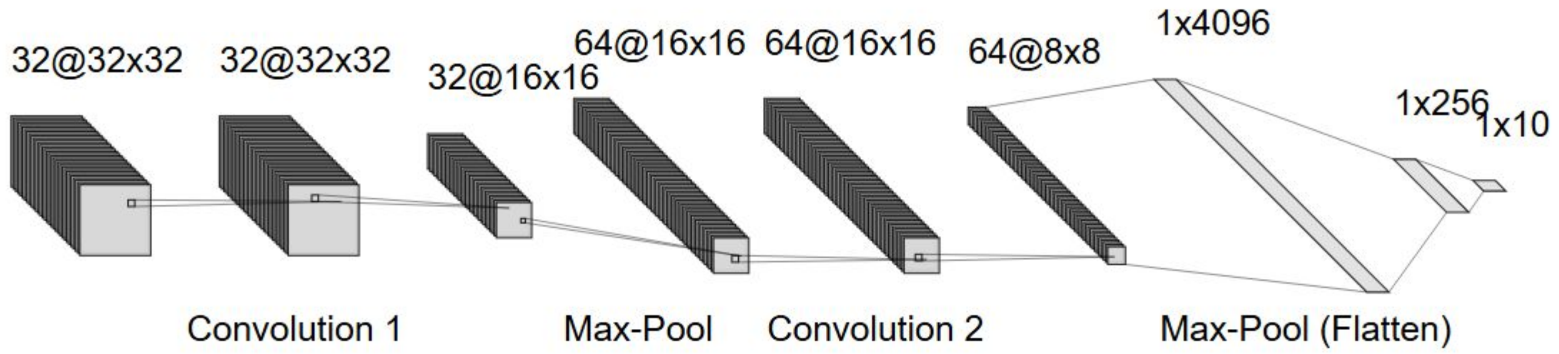
- Analyzes orientation of color gradients in an image
- Extracts rough shapes as features for training



scikit-image team. (n.d.). *Histogram of oriented gradients#*. Histogram of Oriented Gradients. https://scikit-image.org/docs/0.25.x/auto_examples/features_detection/plot_hog.html#id2

CNN Model

- Based on VGG architecture
- Convolution is done in blocks, with multiple passes
- Our model is smaller than most VGG models due to small image size



Hyperparameters

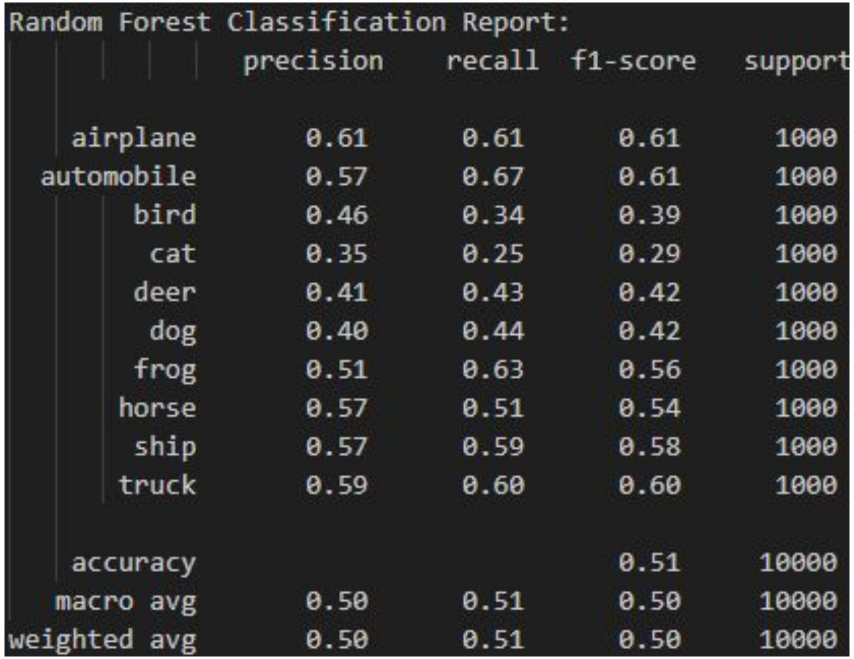
- Traditional models were not tuned, as we want to use them as a baseline for comparing to our CNN
- Any parameters not specified here were left as default values
- Random Forest:
  - n_estimators: 100
- SVC:
  - Linear Kernel (used LinearSVC for faster training)
  - C: 1.00
  - max_iter: 5000
- CNN params were optimized using Adam optimizer
  - Adaptive Moment Estimation
  - Calculate individual learning rates for different parameters
  - Learning rates adapt based on gradients

Results

Random Forest Results

- Highest Precision: Airplane
- Lowest Precision: Cat
- Highest Recall: Automobile
- Lowest Recall: Cat
- Highest F1-Score: Airplane and Automobile
- Lowest F1-Score: Cat
- Overall Accuracy: 51%

Precision = TP / (TP + FP)
Recall = TP / (TP + FN)
F1-Score = 2 x (Precision x Recall)/(Precision + Recall)



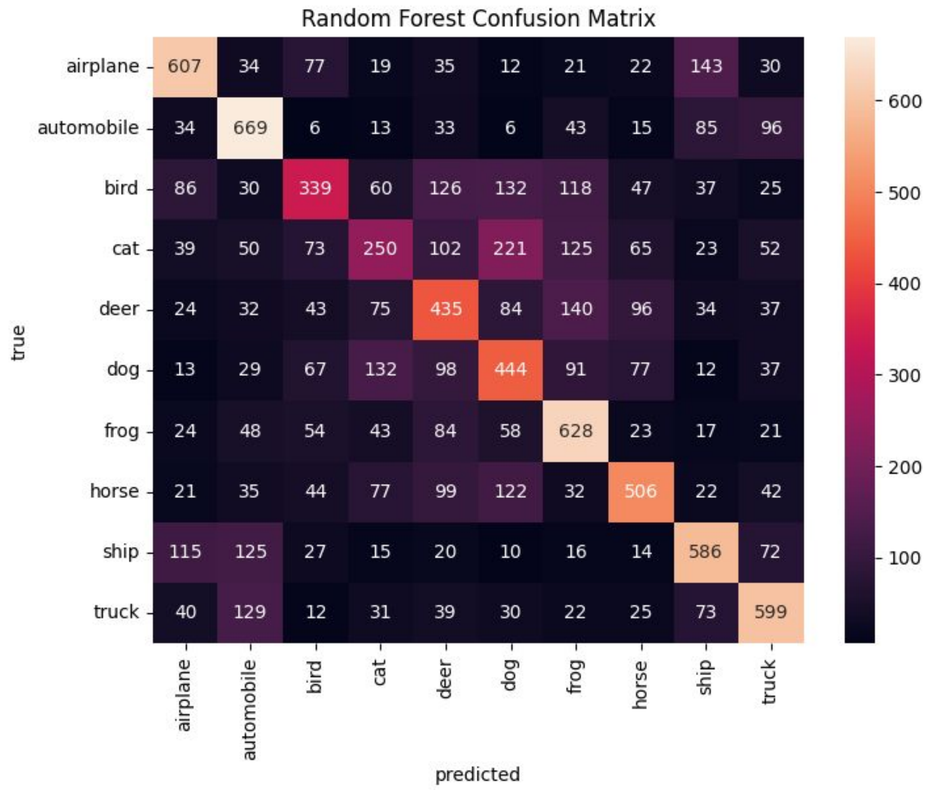
```
Random Forest Classification Report:
              precision    recall  f1-score   support

    airplane       0.61      0.61      0.61      1000
  automobile       0.57      0.67      0.61      1000
        bird       0.46      0.34      0.39      1000
         cat       0.35      0.25      0.29      1000
        deer       0.41      0.43      0.42      1000
         dog       0.40      0.44      0.42      1000
        frog       0.51      0.63      0.56      1000
       horse       0.57      0.51      0.54      1000
        ship       0.57      0.59      0.58      1000
       truck       0.59      0.60      0.60      1000

    accuracy                           0.51     10000
   macro avg       0.50      0.51      0.50     10000
weighted avg       0.50      0.51      0.50     10000
```

Random Forest Results

- Performed fairly poorly
- Really struggled with animals
- Did fairly well with non-organic objects

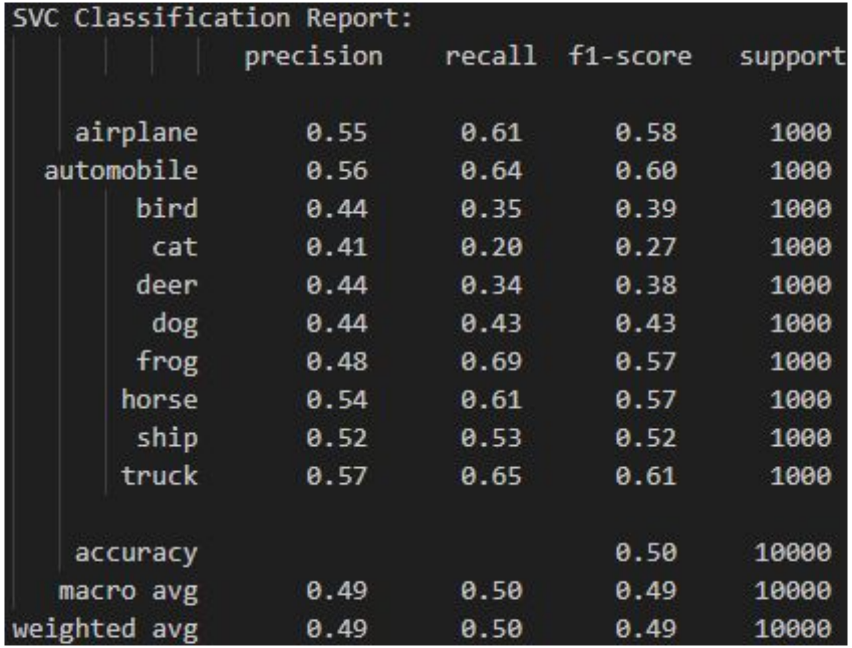


true \ predicted	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
airplane	607	34	77	19	35	12	21	22	143	30
automobile	34	669	6	13	33	6	43	15	85	96
bird	86	30	339	60	126	132	118	47	37	25
cat	39	50	73	250	102	221	125	65	23	52
deer	24	32	43	75	435	84	140	96	34	37
dog	13	29	67	132	98	444	91	77	12	37
frog	24	48	54	43	84	58	628	23	17	21
horse	21	35	44	77	99	122	32	506	22	42
ship	115	125	27	15	20	10	16	14	586	72
truck	40	129	12	31	39	30	22	25	73	599

SVC Results

- Highest Precision: Automobile
- Lowest Precision: Cat
- Highest Recall: Frog
- Lowest Recall: Cat
- Highest F1-Score: Truck
- Lowest F1-Score: Cat
- Overall Accuracy: 50%

Precision = TP / (TP + FP)
Recall = TP / (TP + FN)
F1-Score = 2 x (Precision x Recall)/(Precision + Recall)



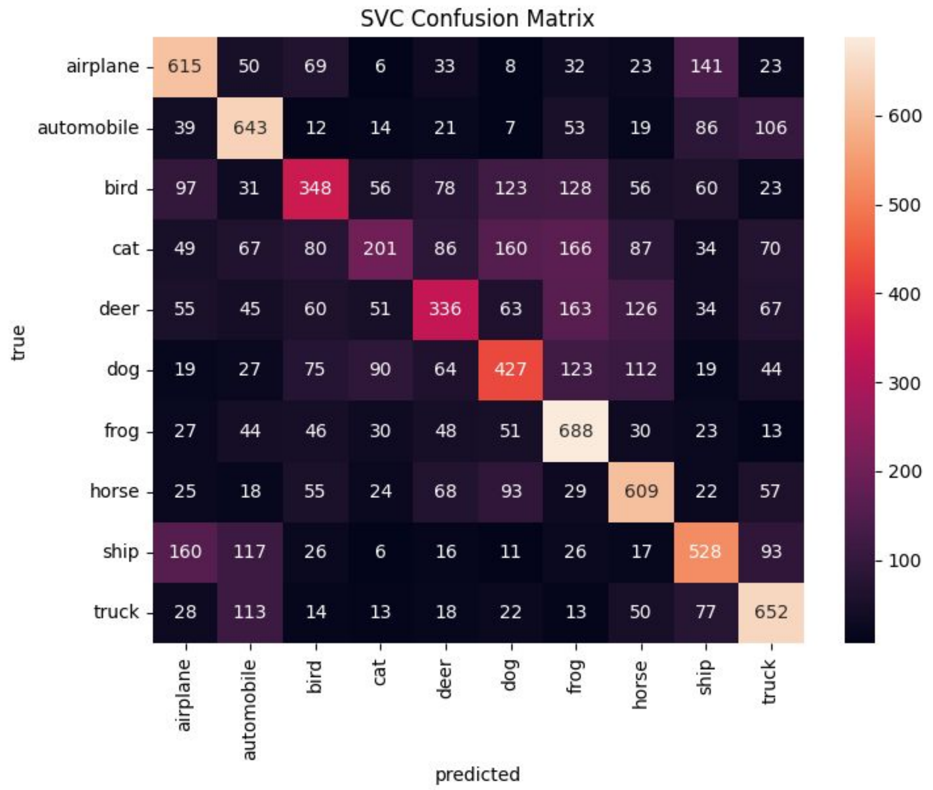
```
SVC Classification Report:
              precision    recall  f1-score   support

    airplane       0.55      0.61      0.58      1000
  automobile       0.56      0.64      0.60      1000
        bird       0.44      0.35      0.39      1000
         cat       0.41      0.20      0.27      1000
        deer       0.44      0.34      0.38      1000
         dog       0.44      0.43      0.43      1000
        frog       0.48      0.69      0.57      1000
       horse       0.54      0.61      0.57      1000
        ship       0.52      0.53      0.52      1000
       truck       0.57      0.65      0.61      1000

    accuracy                           0.50     10000
   macro avg       0.49      0.50      0.49     10000
weighted avg       0.49      0.50      0.49     10000
```

SVC Results

- Performed fairly poorly
- Struggled with animals as well
- Did okay with non-organic objects

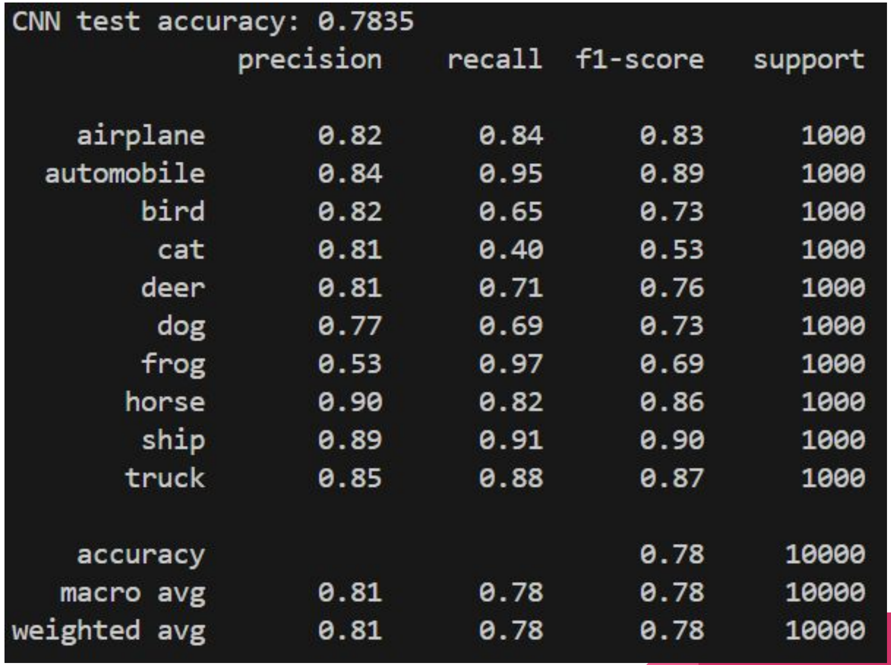


true \ predicted	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
airplane	615	50	69	6	33	8	32	23	141	23
automobile	39	643	12	14	21	7	53	19	86	106
bird	97	31	348	56	78	123	128	56	60	23
cat	49	67	80	201	86	160	166	87	34	70
deer	55	45	60	51	336	63	163	126	34	67
dog	19	27	75	90	64	427	123	112	19	44
frog	27	44	46	30	48	51	688	30	23	13
horse	25	18	55	24	68	93	29	609	22	57
ship	160	117	26	6	16	11	26	17	528	93
truck	28	113	14	13	18	22	13	50	77	652

CNN Results

- Highest Precision: Horse
- Lowest Precision: Frog
- Highest Recall: Frog
- Lowest Recall: Cat
- Highest F1-Score: Ship
- Lowest F1-Score: Cat
- Overall Accuracy: 78%

Precision = TP / (TP + FP)
Recall = TP / (TP + FN)
F1-Score = 2 x (Precision x Recall)/(Precision + Recall)



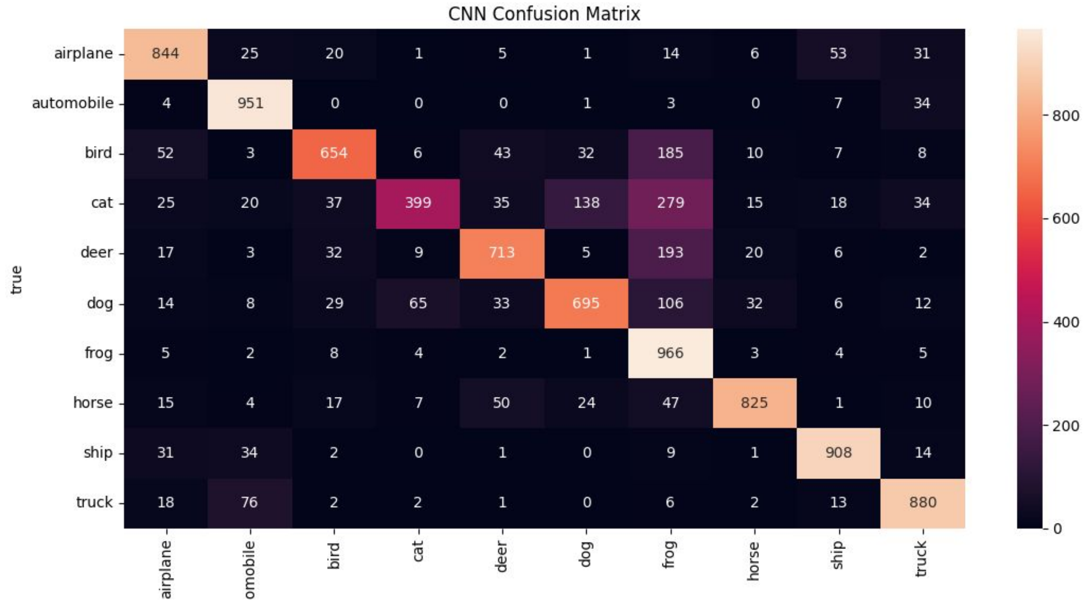
```
CNN test accuracy: 0.7835
              precision    recall  f1-score   support

    airplane       0.82      0.84      0.83      1000
  automobile       0.84      0.95      0.89      1000
        bird       0.82      0.65      0.73      1000
         cat       0.81      0.40      0.53      1000
        deer       0.81      0.71      0.76      1000
         dog       0.77      0.69      0.73      1000
        frog       0.53      0.97      0.69      1000
       horse       0.90      0.82      0.86      1000
        ship       0.89      0.91      0.90      1000
       truck       0.85      0.88      0.87      1000

    accuracy                           0.78     10000
   macro avg       0.81      0.78      0.78     10000
weighted avg       0.81      0.78      0.78     10000
```

CNN Results

- Possible overfit with frog classification
- Low cat accuracy
- Fairly good results



	airplane	omobile	bird	cat	deer	dog	frog	horse	ship	truck
airplane	844	25	20	1	5	1	14	6	53	31
automobile	4	951	0	0	0	1	3	0	7	34
bird	52	3	654	6	43	32	185	10	7	8
cat	25	20	37	399	35	138	279	15	18	34
deer	17	3	32	9	713	5	193	20	6	2
dog	14	8	29	65	33	695	106	32	6	12
frog	5	2	8	4	2	1	966	3	4	5
horse	15	4	17	7	50	24	47	825	1	10
ship	31	34	2	0	1	0	9	1	908	14
truck	18	76	2	2	1	0	6	2	13	880

Conclusions

Interpretations

- The CNN model is a very accurate model for image classification.
- All of our models had a tendency to predict “frog” on the images, with CNN doing it the most. This is most likely due to some similarities in features between the image classes.
  - It is interesting how the model falsely predicted many classes to be “frog” but did not falsely predict “frog” to be those classes.
- All of our models performed poorly with the “cat” classification. “Cat” had the lowest recall and f1-score in every model.

Comparison

- Our CNN model performed very well, and predictably better than the traditional models
- The traditional models both performed similarly to each other, with little to noticeable differences
  - The SVC performed slightly worse with non-organic objects

Limitations

- Computing power was a huge limitation on our model complexity
- The dataset itself could be giving us worse results. Many of the image classes have similar color schemes and shapes, which we believe to be a reason why the model is predicting “frog” very often.
    - Many of the animal classes have similar color schemes due to the tendency of animals being found in nature

Future Work

- Could try testing model on a different dataset of the same image classes.
- More resources would lead to being able to use RBF kernel for SVC and potentially a better comparison
- The complexity of our CNN could also be increased with more computing power
- Increase resolution of images or add more data augmentation.

Any Questions?