Ship Image Classification using Deep Learning

Ahsan Khan

Introduction to Deep Learning

University of Colorado Boulder

Dataset Overview

The dataset consists of images of various ship types, categorized into five classes:

- Cargo (1)
- Military (2)
- Carrier (3)
- Cruise (4)
- Tankers (5)

The dataset includes 6,252 images for training and 2,680 images for testing.

Project Objective

To develop and compare different deep learning models for accurate ship classification

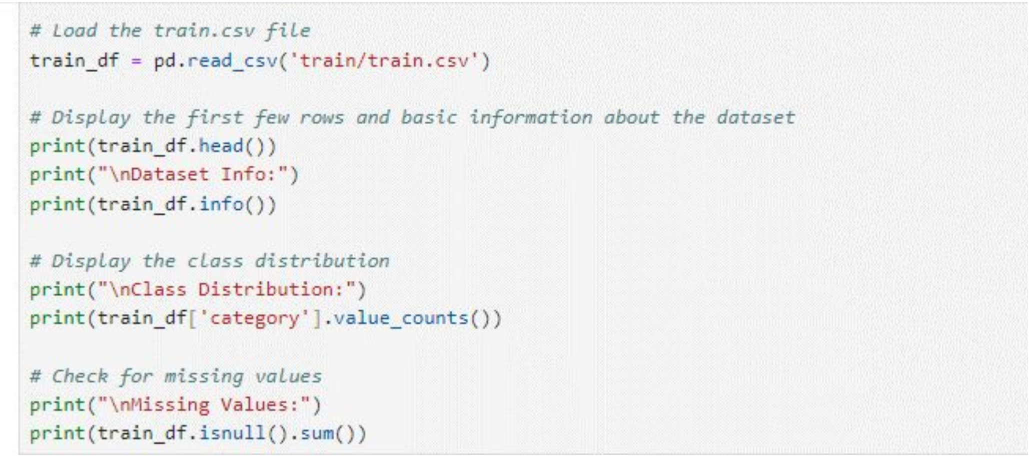
Selecting the best performing model for potential real-world applications.

Create a scalable solution for high-volume data processing

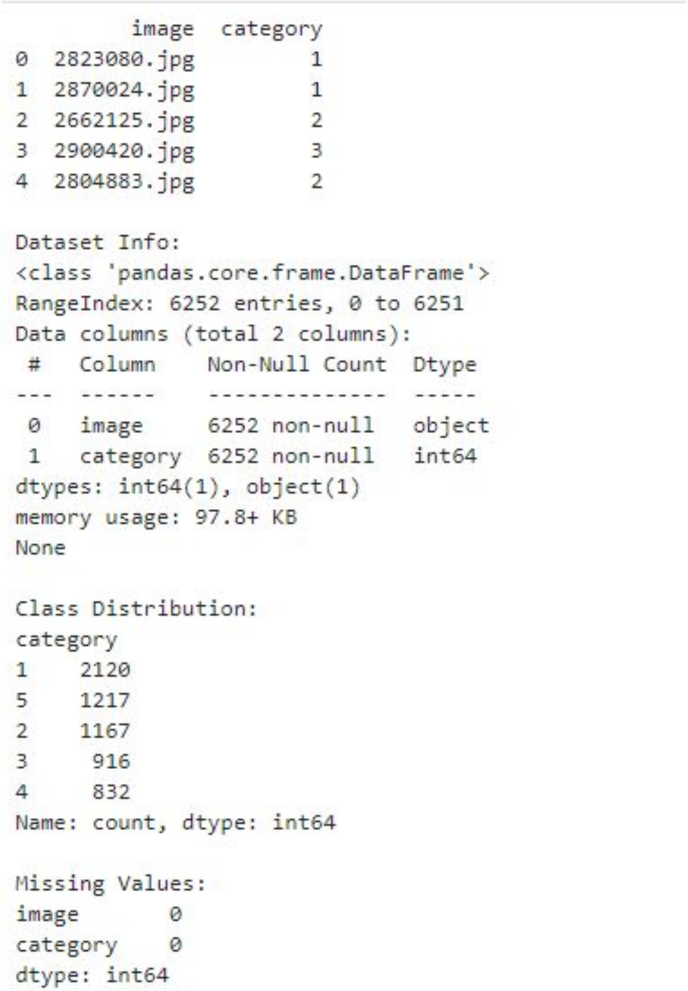
Methodology

- Data Loading and Initial Exploration
- Exploratory Data Analysis (EDA)
- Data Preprocessing
- Model Development
- Model Comparison
- Hyperparameter Tuning
- Comparison and Evaluation of Results
- Conclusion

Data Loading and Initial Exploration

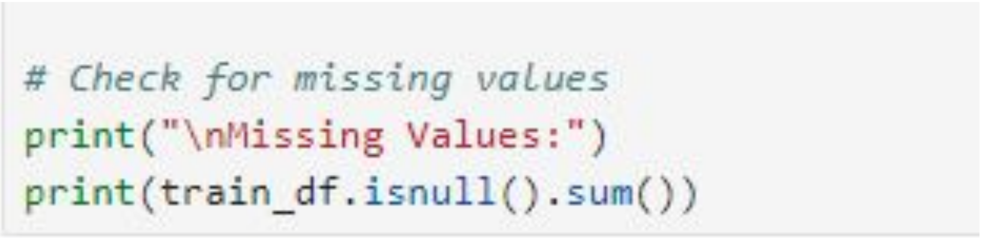


```python
# Load the train.csv file
train_df = pd.read_csv('train/train.csv')

# Display the first few rows and basic information about the dataset
print(train_df.head())
print("\nDataset Info:")
print(train_df.info())

# Display the class distribution
print("\nClass Distribution:")
print(train_df['category'].value_counts())

# Check for missing values
print("\nMissing Values:")
print(train_df.isnull().sum())
```

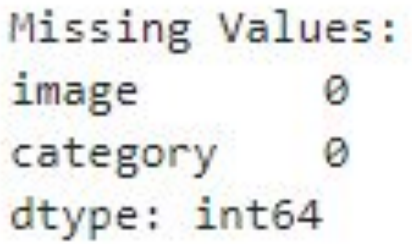


```
         image  category
0  2823080.jpg         1
1  2870024.jpg         1
2  2662125.jpg         2
3  2900420.jpg         3
4  2804883.jpg         2

Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6252 entries, 0 to 6251
Data columns (total 2 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   image     6252 non-null   object
 1   category  6252 non-null   int64 
dtypes: int64(1), object(1)
memory usage: 97.8+ KB
None

Class Distribution:
category
1    2120
5    1217
2    1167
3     916
4     832
Name: count, dtype: int64

Missing Values:
image       0
category    0
dtype: int64
```

Data Cleaning

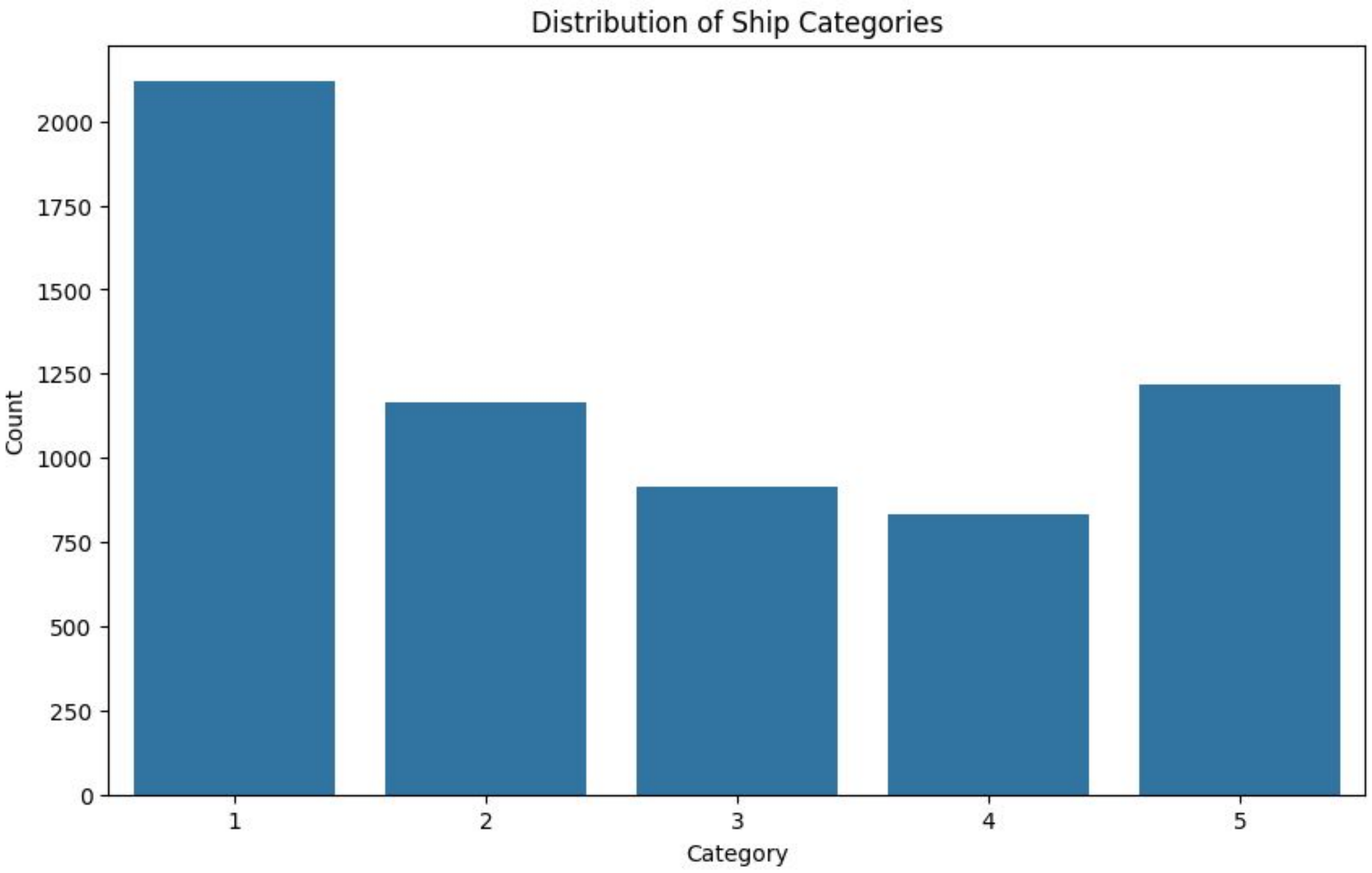


```
# Check for missing values
print("\nMissing Values:")
print(train_df.isnull().sum())
```

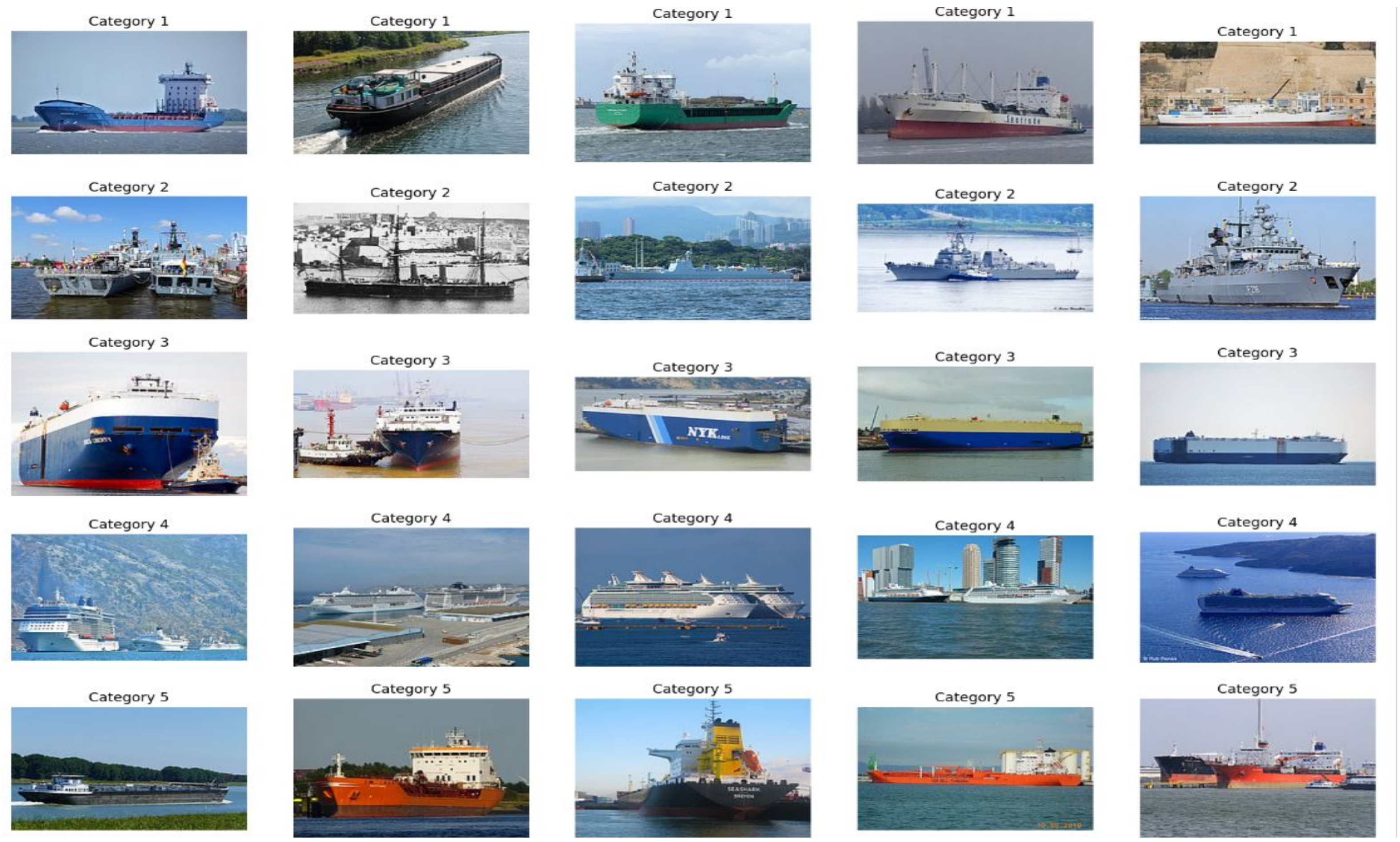


```
Missing Values:
image       0
category    0
dtype: int64
```

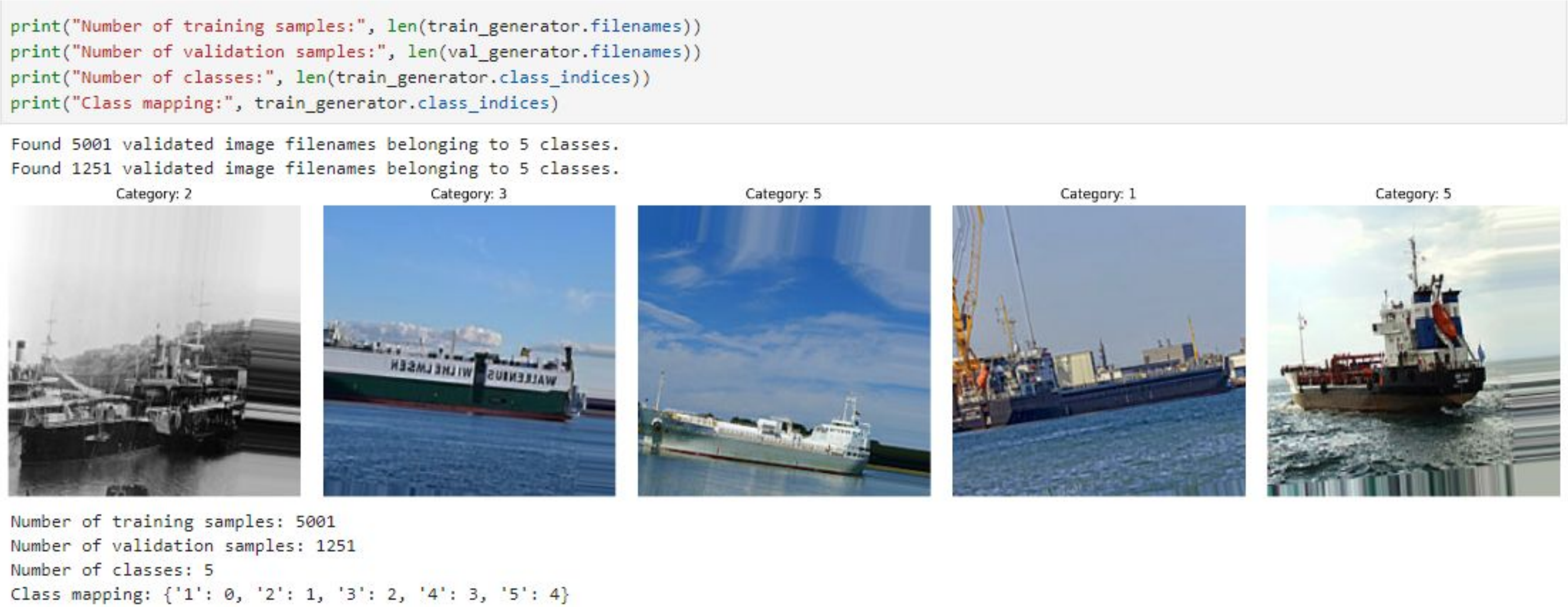
Exploratory Data Analysis (EDA)



Exploratory Data Analysis (EDA)



Data Preprocessing

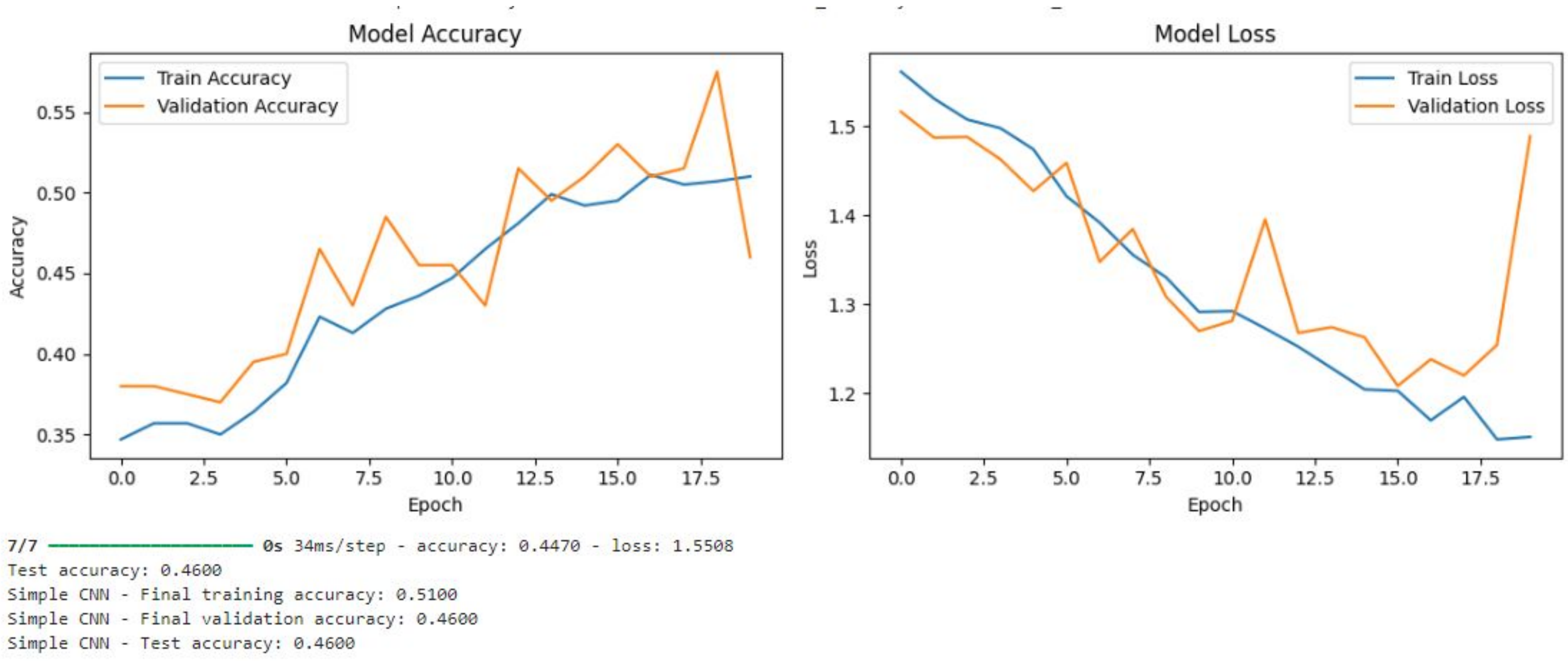


```
print("Number of training samples:", len(train_generator.filenames))
print("Number of validation samples:", len(val_generator.filenames))
print("Number of classes:", len(train_generator.class_indices))
print("Class mapping:", train_generator.class_indices)
```

```
Found 5001 validated image filenames belonging to 5 classes.
Found 1251 validated image filenames belonging to 5 classes.
```

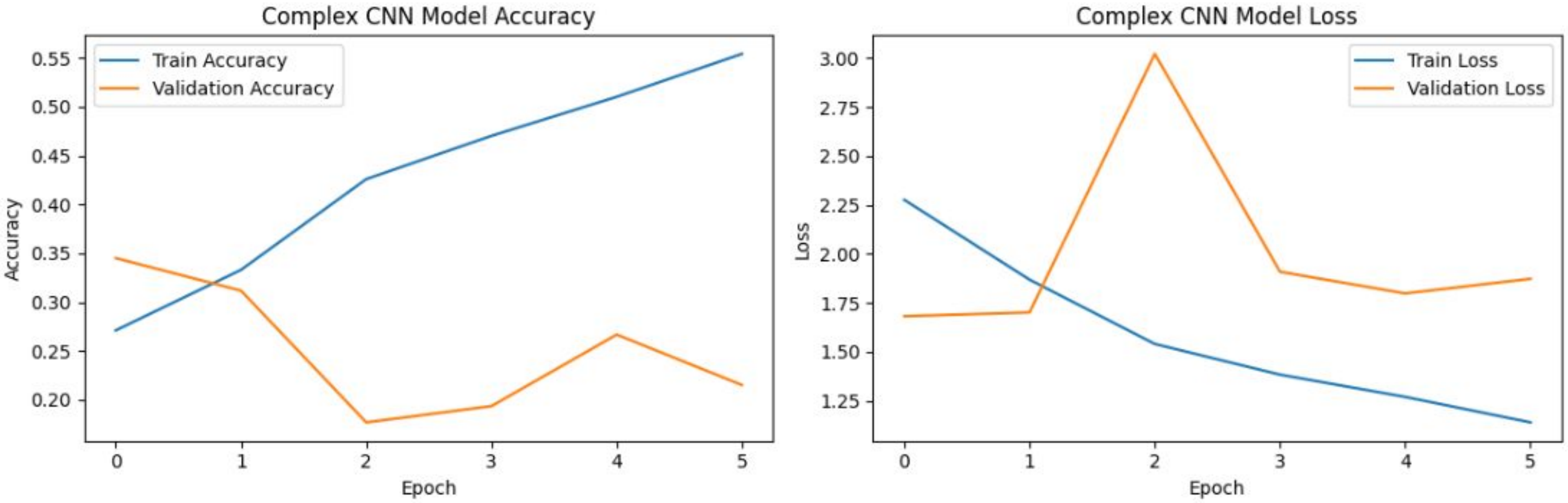
```
Number of training samples: 5001
Number of validation samples: 1251
Number of classes: 5
Class mapping: {'1': 0, '2': 1, '3': 2, '4': 3, '5': 4}
```

Simple CNN Model



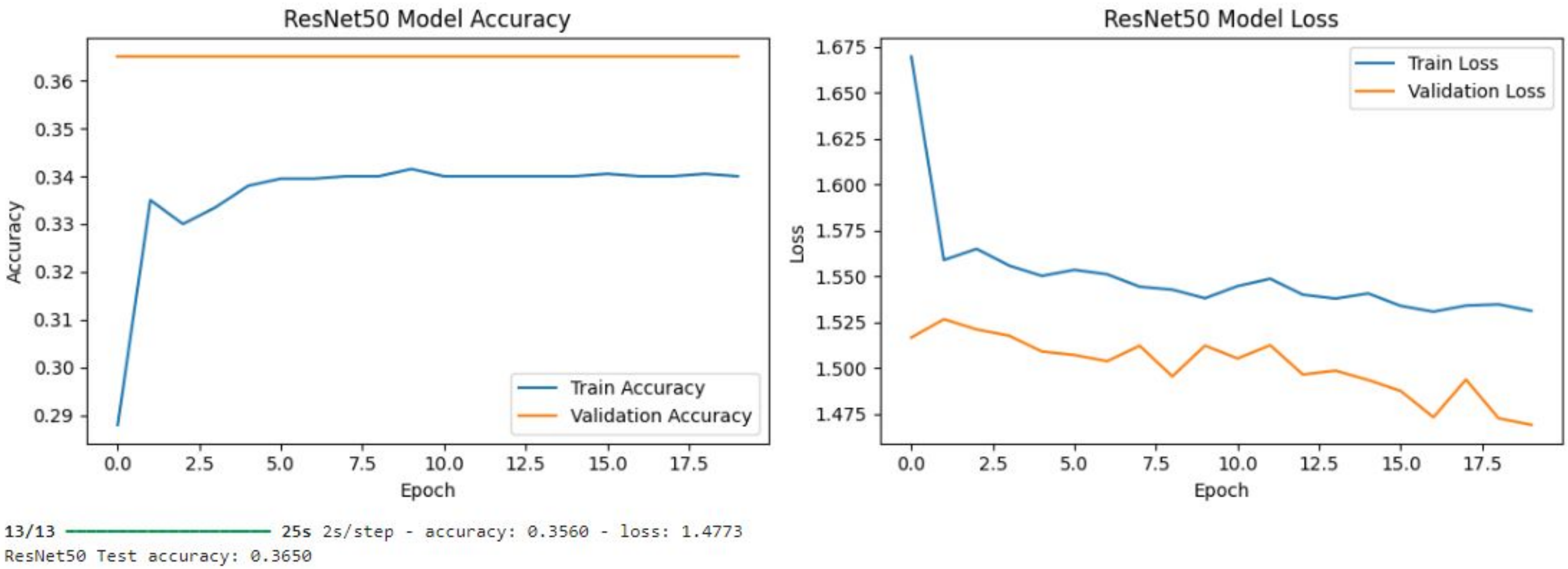
```
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step - accuracy: 0.4470 - loss: 1.5508
Test accuracy: 0.4600
Simple CNN - Final training accuracy: 0.5100
Simple CNN - Final validation accuracy: 0.4600
Simple CNN - Test accuracy: 0.4600
```

Complex CNN Model



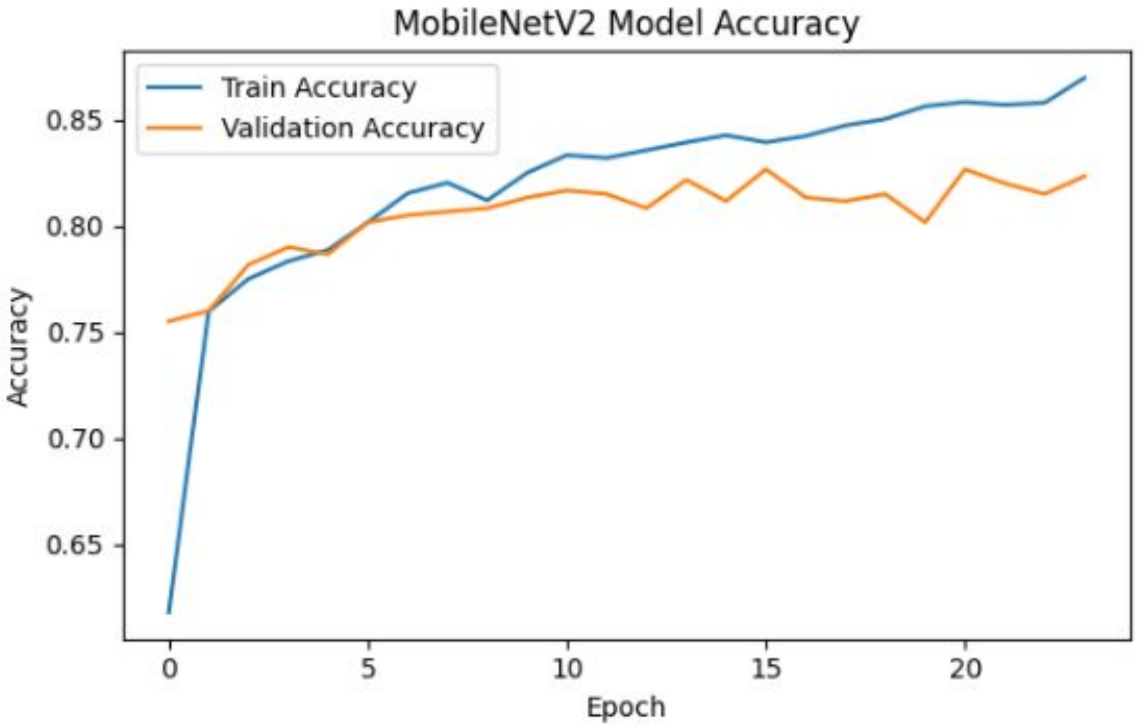
```
10/10 ━━━━━━━━━━━━━━━━━━━━ 3s 266ms/step - accuracy: 0.3596 - loss: 1.6528
Complex CNN Test accuracy: 0.3450
```

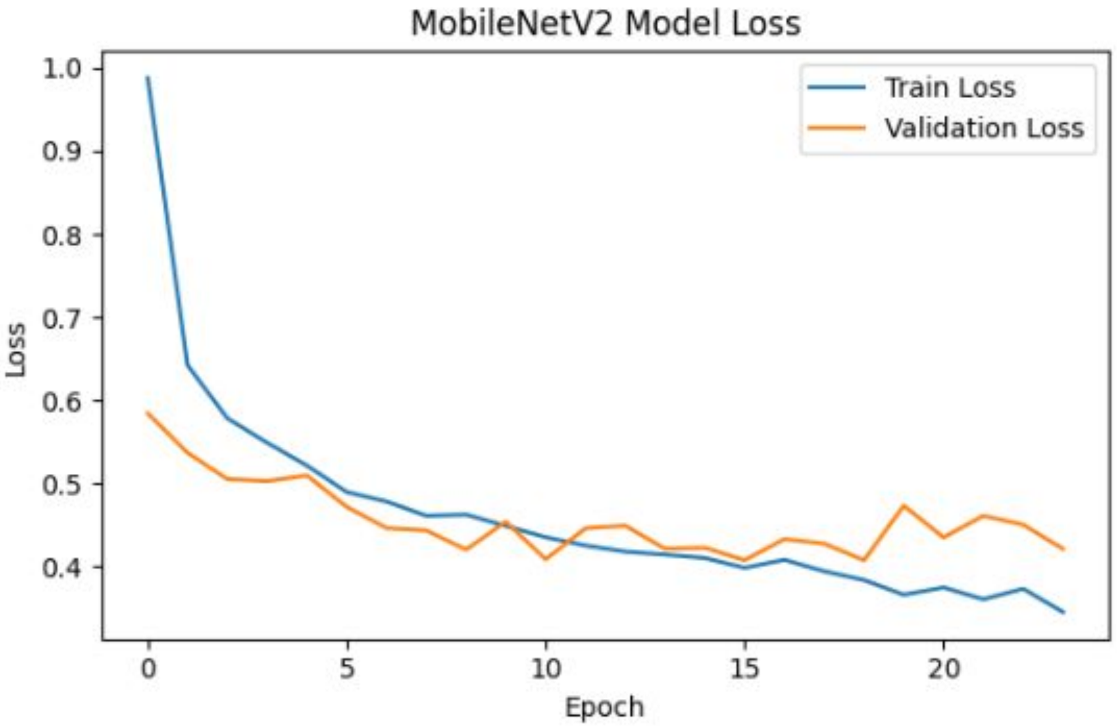
ResNet50 Model

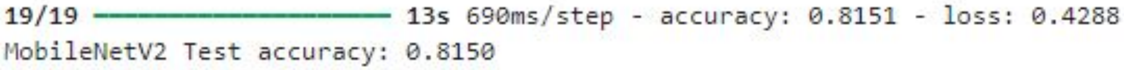


```
13/13 ━━━━━━━━━━━━━━━━━━━━ 25s 2s/step - accuracy: 0.3560 - loss: 1.4773
ResNet50 Test accuracy: 0.3650
```

MobileNetV2 Model

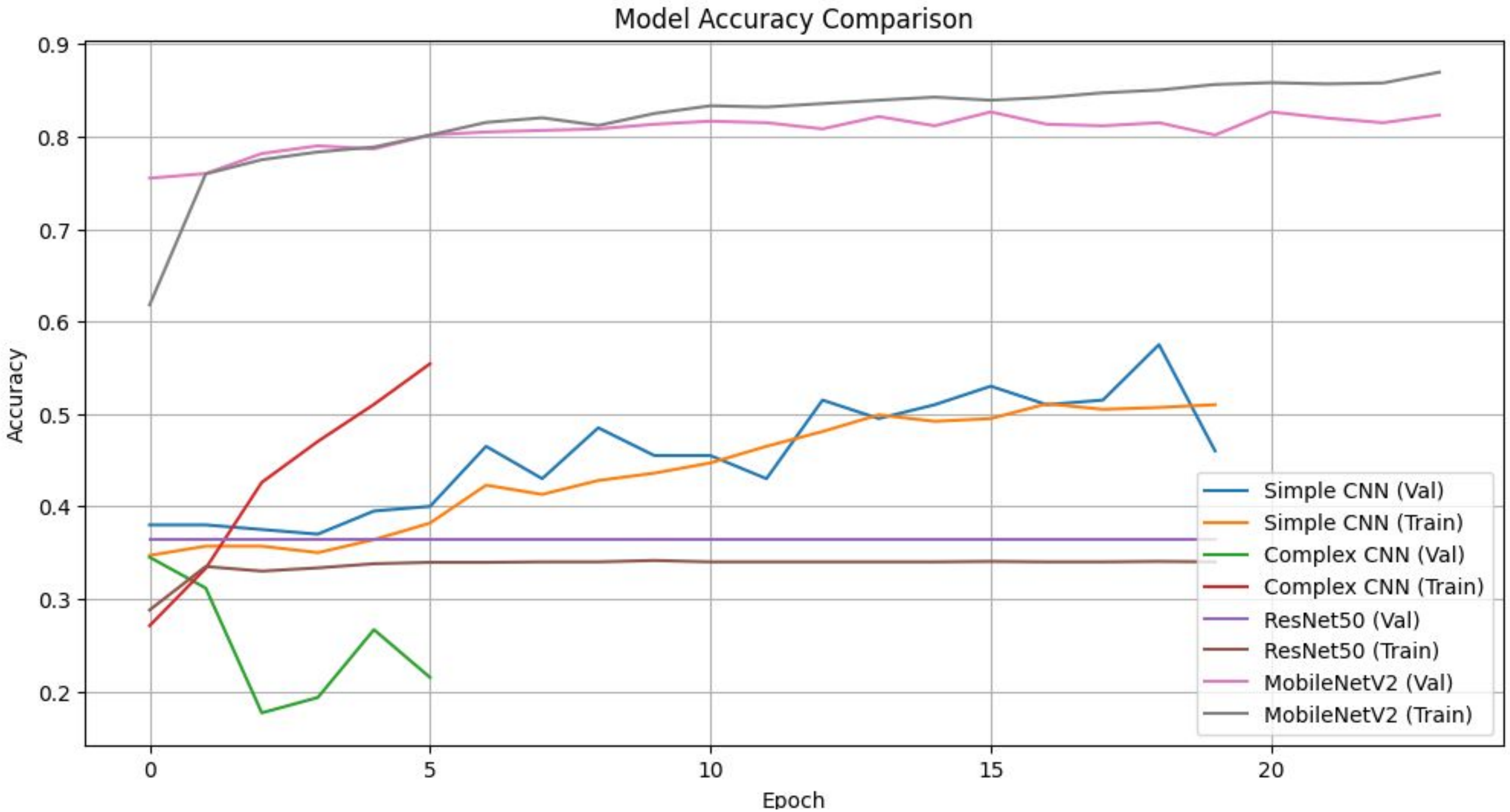




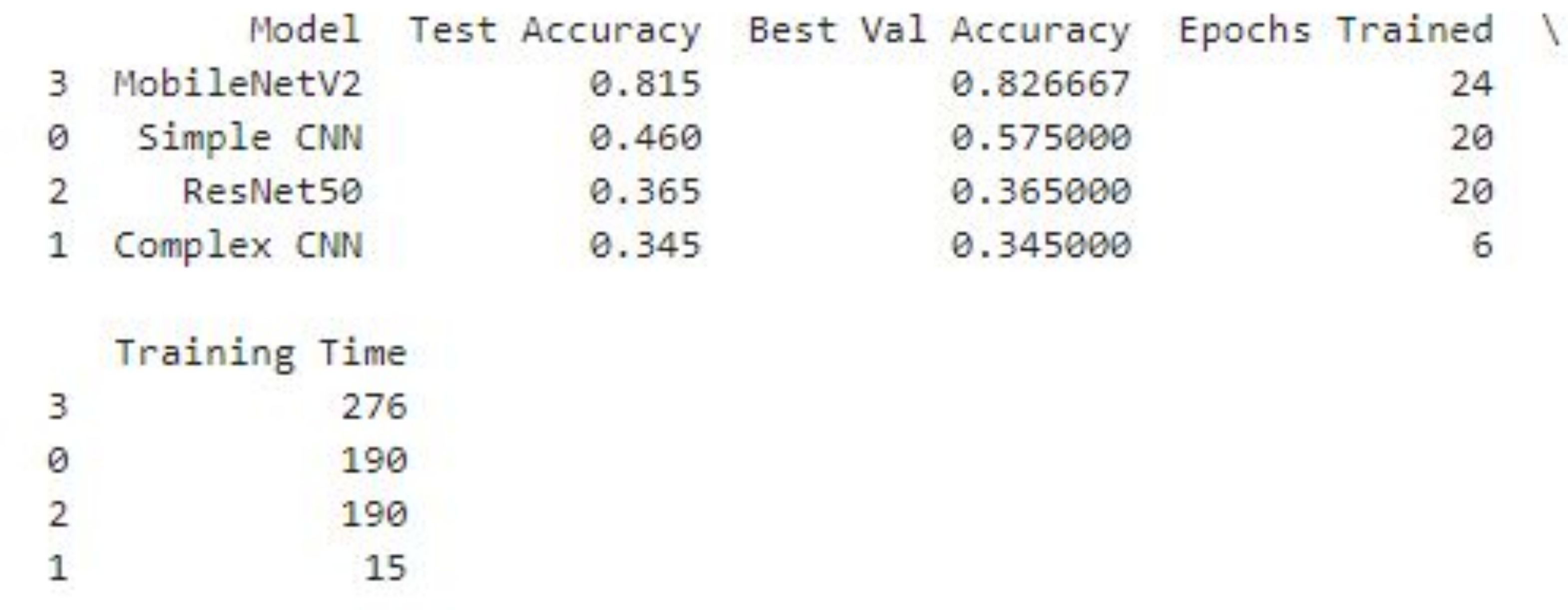


```
19/19 ━━━━━━━━━━━━━━━━━━━━ 13s 690ms/step - accuracy: 0.8151 - loss: 0.4288
MobileNetV2 Test accuracy: 0.8150
```

Model Accuracy Comparison

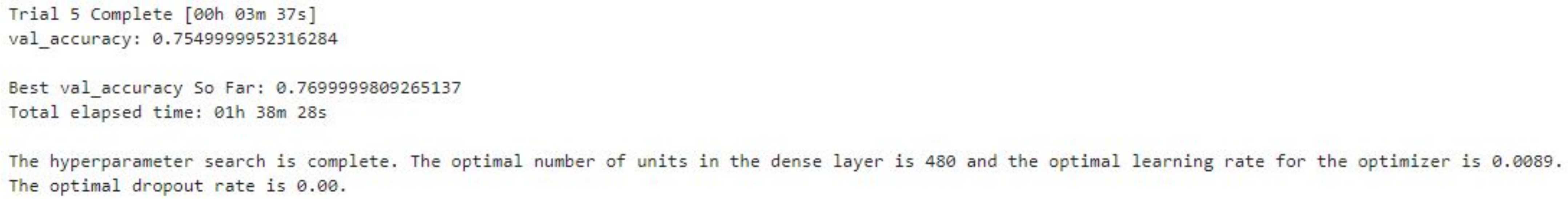


Model Comparison Summary



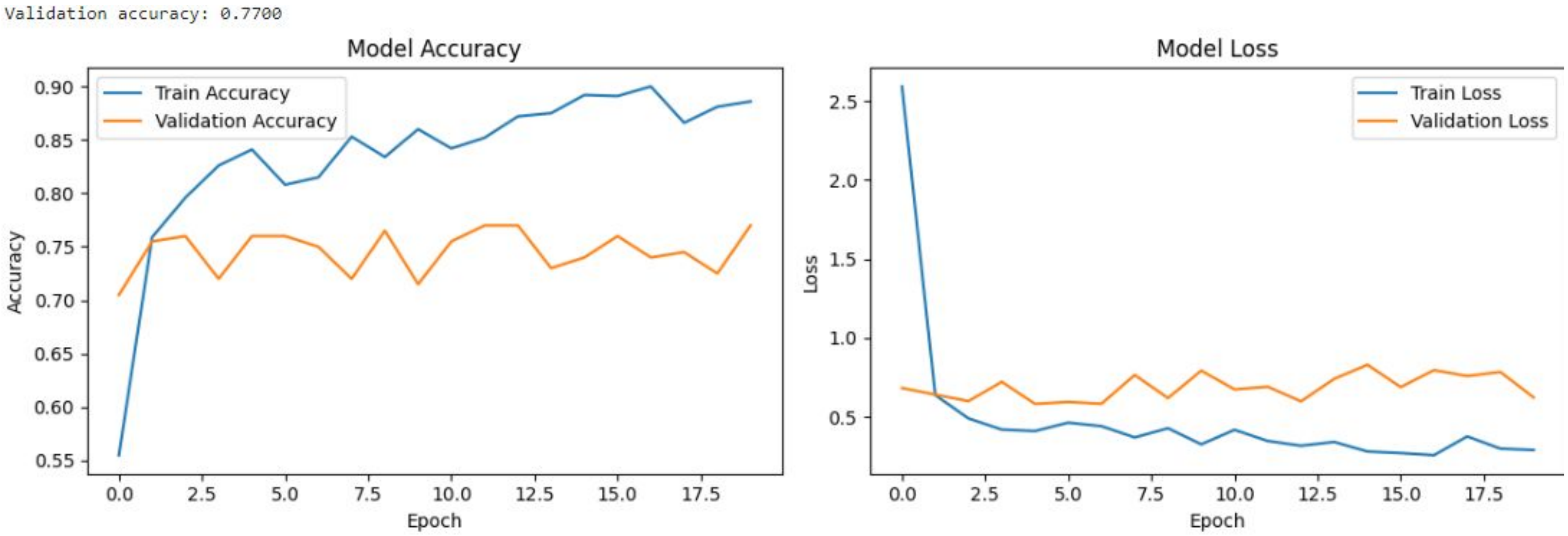
```
          Model  Test Accuracy  Best Val Accuracy  Epochs Trained  \
3  MobileNetV2          0.815           0.826667              24   
0   Simple CNN          0.460           0.575000              20   
2     ResNet50          0.365           0.365000              20   
1  Complex CNN          0.345           0.345000               6   

   Training Time  
3            276  
0            190  
2            190  
1             15  
```

Hyperparameter Tuning



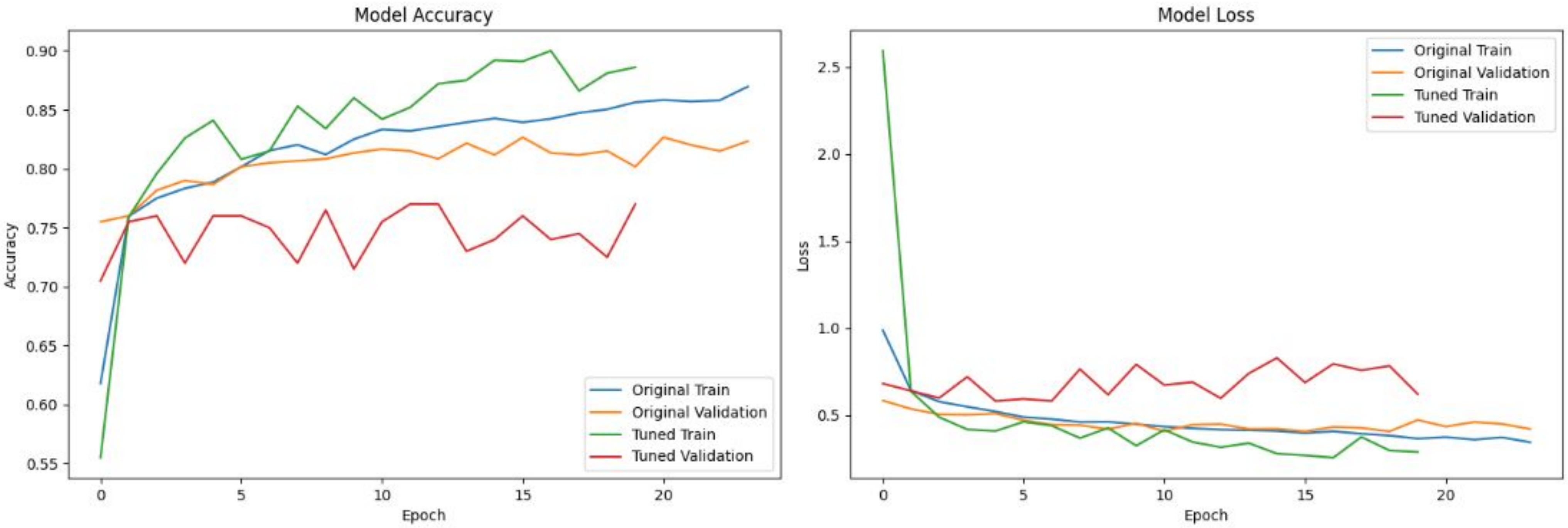
```
Trial 5 Complete [00h 03m 37s]
val_accuracy: 0.7549999952316284

Best val_accuracy So Far: 0.7699999809265137
Total elapsed time: 01h 38m 28s

The hyperparameter search is complete. The optimal number of units in the dense layer is 480 and the optimal learning rate for the optimizer is 0.0089.
The optimal dropout rate is 0.00.
```

Hyperparameter Tuning

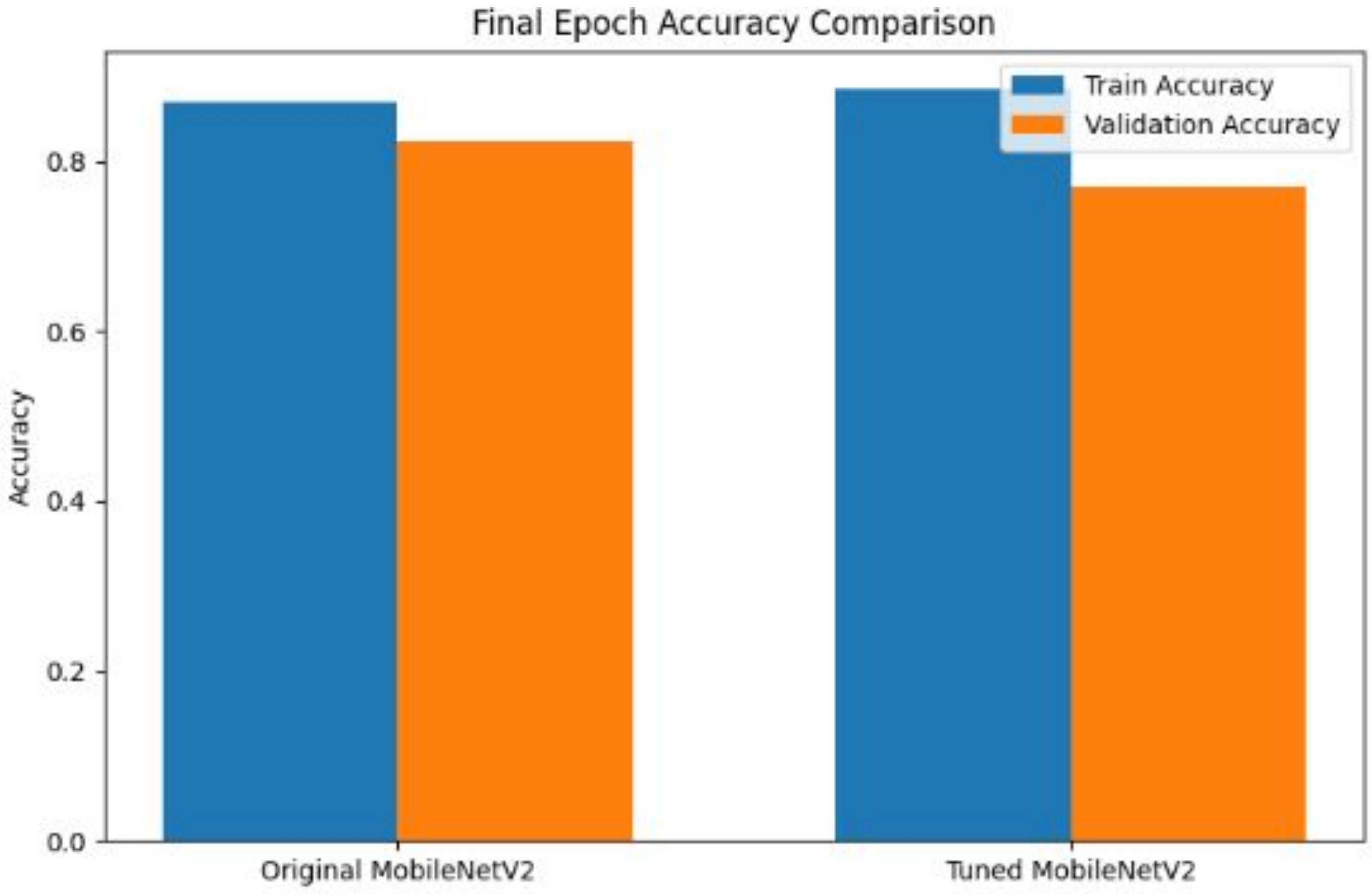


```
Validation accuracy: 0.7700
```

Hyperparameter Tuning Comparison



Hyperparameter Tuning Comparison



Conclusion

- The MobileNetV2 model demonstrated the best balance between performance and efficiency for this ship classification task.
- Transfer learning proved highly effective, significantly outperforming custom-built CNN architectures.
- The project highlighted the importance of model selection and the potential of lightweight architectures like MobileNetV2 for practical applications.