Model_Scratch 01 - No data augmentation

Name: Alberto Pingo
Email: 2202145 @my.ipleiria.pt

Validation dataset: train5

Directories

This section sets up the directory paths used for training, validation, and test datasets based on the repository structure.

In [ ]:
```
import os

current_dir = os.getcwd()

# TWO FOLDERS UP
data_dir = os.path.abspath(os.path.join(current_dir, os.pardir, os.pardir
test_dir = os.path.join(data_dir, 'test')
train_dir = os.path.join(data_dir, 'train')

train_dirs = []
for i in range(1, 5):
    train_dirs.append(os.path.join(train_dir, 'train' + str(i)))

validation_dir = os.path.join(data_dir, 'train', 'train5')

print(current_dir)
print(data_dir)
print(test_dir)
print(train_dir)
print(validation_dir)
```

```
/home/pws/code/IA-image-classification/notebooks/models-S
/home/pws/code/IA-image-classification/data
/home/pws/code/IA-image-classification/data/test
/home/pws/code/IA-image-classification/data/train
/home/pws/code/IA-image-classification/data/train/train5
```

Preprocessing

Load the datasets and perform initial preprocessing. Images are resized to 32x32 pixels and batched.

In [ ]:
```
from keras.utils import image_dataset_from_directory
import tensorflow as tf

# Load training datasets from train1 to train4
train_datasets = []
IMG_SIZE = 32
```

```
BATCH_SIZE = 64
for i in range(1, 5):
    dataset = image_dataset_from_directory(train_dirs[i-1], image_size=(I
    train_datasets.append(dataset)

train_dataset = train_datasets[0]
for dataset in train_datasets[1:]:
    train_dataset = train_dataset.concatenate(dataset)

validation_dataset = image_dataset_from_directory(validation_dir, image_s


test_dataset = image_dataset_from_directory(test_dir, image_size=(IMG_SIZ

class_names = validation_dataset.class_names
class_names = [class_name.split('_')[-1] for class_name in class_names]

print(class_names)
```

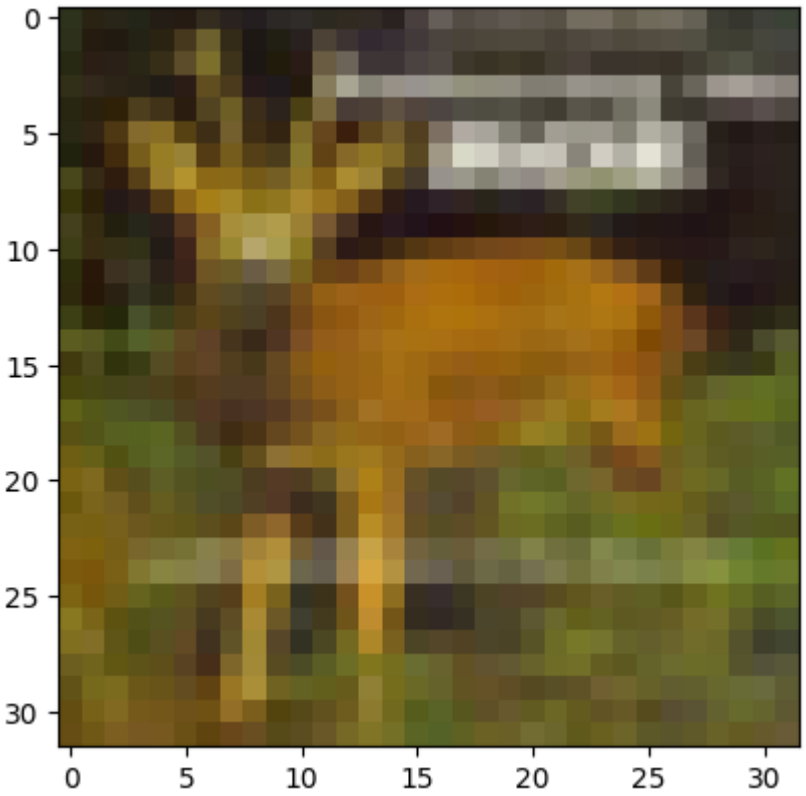
```
Found 10000 files belonging to 10 classes.
```

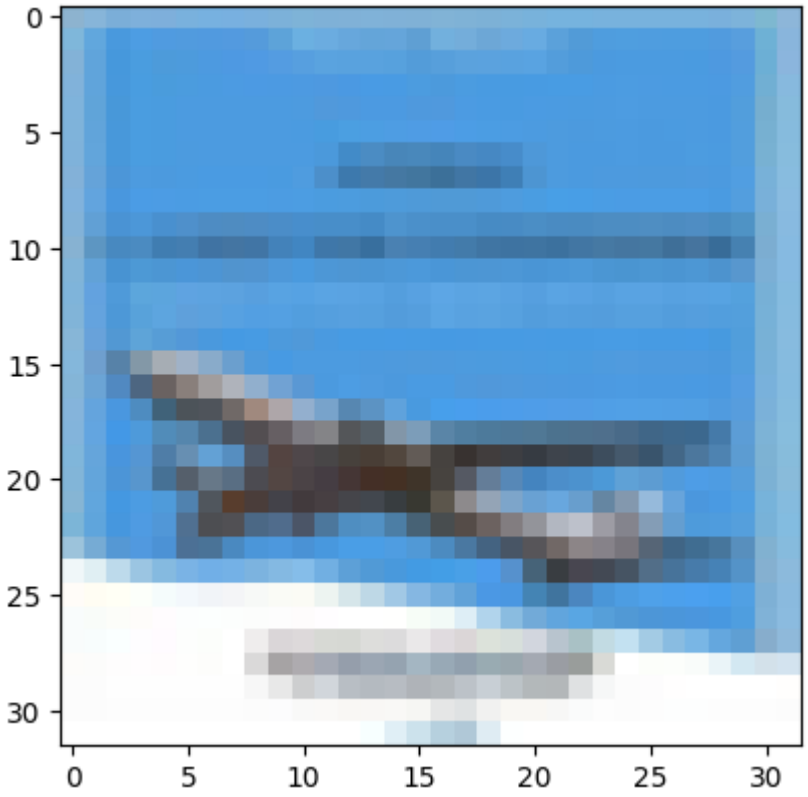
```
2024-06-22 20:35:54.599598: I external/local_xla/xla/stream_executor/cuda/
cuda_executor.cc:901] successful NUMA node read from SysFS had negative va
lue (-1), but there must be at least one NUMA node, so returning NUMA node
zero. See more at https://github.com/torvalds/linux/blob/v6.0/Documentatio
n/ABI/testing/sysfs-bus-pci#L344-L355
2024-06-22 20:35:55.059475: I external/local_xla/xla/stream_executor/cuda/
cuda_executor.cc:901] successful NUMA node read from SysFS had negative va
lue (-1), but there must be at least one NUMA node, so returning NUMA node
zero. See more at https://github.com/torvalds/linux/blob/v6.0/Documentatio
n/ABI/testing/sysfs-bus-pci#L344-L355
2024-06-22 20:35:55.059653: I external/local_xla/xla/stream_executor/cuda/
cuda_executor.cc:901] successful NUMA node read from SysFS had negative va
lue (-1), but there must be at least one NUMA node, so returning NUMA node
zero. See more at https://github.com/torvalds/linux/blob/v6.0/Documentatio
n/ABI/testing/sysfs-bus-pci#L344-L355
2024-06-22 20:35:55.060425: I external/local_xla/xla/stream_executor/cuda/
cuda_executor.cc:901] successful NUMA node read from SysFS had negative va
lue (-1), but there must be at least one NUMA node, so returning NUMA node
zero. See more at https://github.com/torvalds/linux/blob/v6.0/Documentatio
n/ABI/testing/sysfs-bus-pci#L344-L355
2024-06-22 20:35:55.060587: I external/local_xla/xla/stream_executor/cuda/
cuda_executor.cc:901] successful NUMA node read from SysFS had negative va
lue (-1), but there must be at least one NUMA node, so returning NUMA node
zero. See more at https://github.com/torvalds/linux/blob/v6.0/Documentatio
n/ABI/testing/sysfs-bus-pci#L344-L355
2024-06-22 20:35:55.060761: I external/local_xla/xla/stream_executor/cuda/
cuda_executor.cc:901] successful NUMA node read from SysFS had negative va
lue (-1), but there must be at least one NUMA node, so returning NUMA node
zero. See more at https://github.com/torvalds/linux/blob/v6.0/Documentatio
n/ABI/testing/sysfs-bus-pci#L344-L355
2024-06-22 20:35:55.282026: I external/local_xla/xla/stream_executor/cuda/
cuda_executor.cc:901] successful NUMA node read from SysFS had negative va
lue (-1), but there must be at least one NUMA node, so returning NUMA node
zero. See more at https://github.com/torvalds/linux/blob/v6.0/Documentatio
n/ABI/testing/sysfs-bus-pci#L344-L355
2024-06-22 20:35:55.282264: I external/local_xla/xla/stream_executor/cuda/
cuda_executor.cc:901] successful NUMA node read from SysFS had negative va
lue (-1), but there must be at least one NUMA node, so returning NUMA node
zero. See more at https://github.com/torvalds/linux/blob/v6.0/Documentatio
n/ABI/testing/sysfs-bus-pci#L344-L355
2024-06-22 20:35:55.282471: I external/local_xla/xla/stream_executor/cuda/
cuda_executor.cc:901] successful NUMA node read from SysFS had negative va
lue (-1), but there must be at least one NUMA node, so returning NUMA node
zero. See more at https://github.com/torvalds/linux/blob/v6.0/Documentatio
n/ABI/testing/sysfs-bus-pci#L344-L355
2024-06-22 20:35:55.282592: I tensorflow/core/common_runtime/gpu/gpu_devic
e.cc:1929] Created device /job:localhost/replica:0/task:0/device:GPU:0 wit
h 300 MB memory:  -> device: 0, name: NVIDIA GeForce GTX 1060 6GB, pci bus
id: 0000:01:00.0, compute capability: 6.1
Found 10000 files belonging to 10 classes.
Found 10000 files belonging to 10 classes.
Found 10000 files belonging to 10 classes.
Found 10000 files belonging to 10 classes.
Found 10000 files belonging to 10 classes.
['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse',
'ship', 'truck']
```

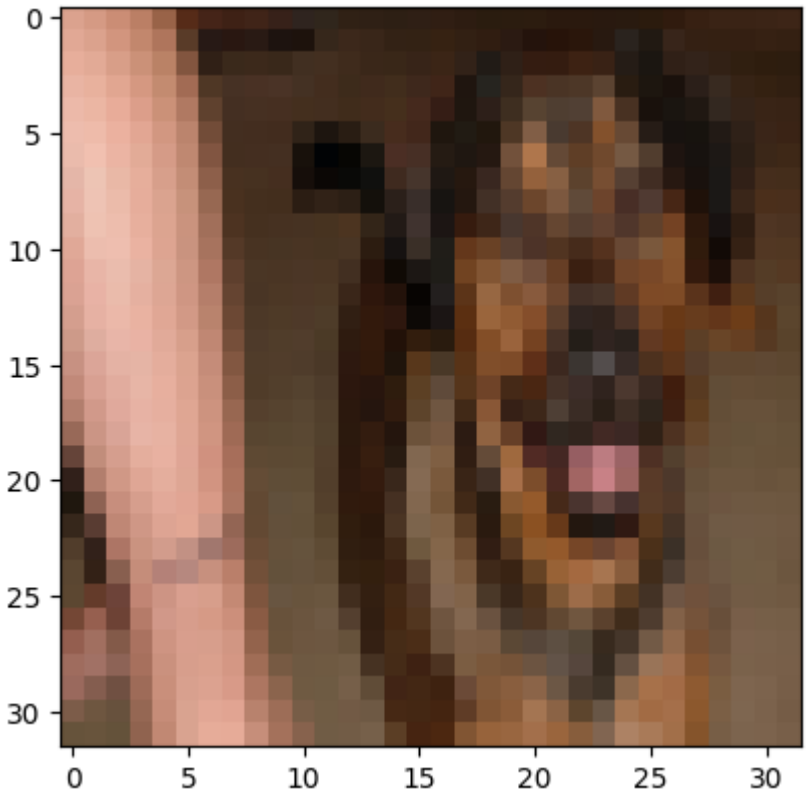
Configure the dataset for performance

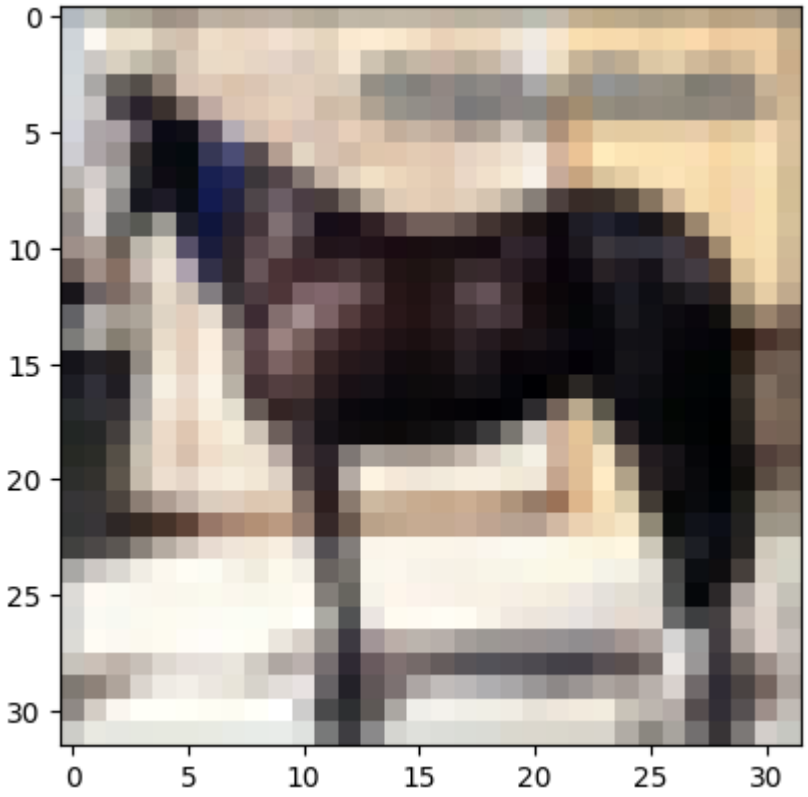
In [ ]:
```
AUTOTUNE = tf.data.AUTOTUNE

train_dataset = train_dataset.cache().shuffle(1000).prefetch(buffer_size=
validation_dataset = validation_dataset.cache().prefetch(buffer_size=AUTO
test_dataset = test_dataset.cache().prefetch(buffer_size=AUTOTUNE)
```

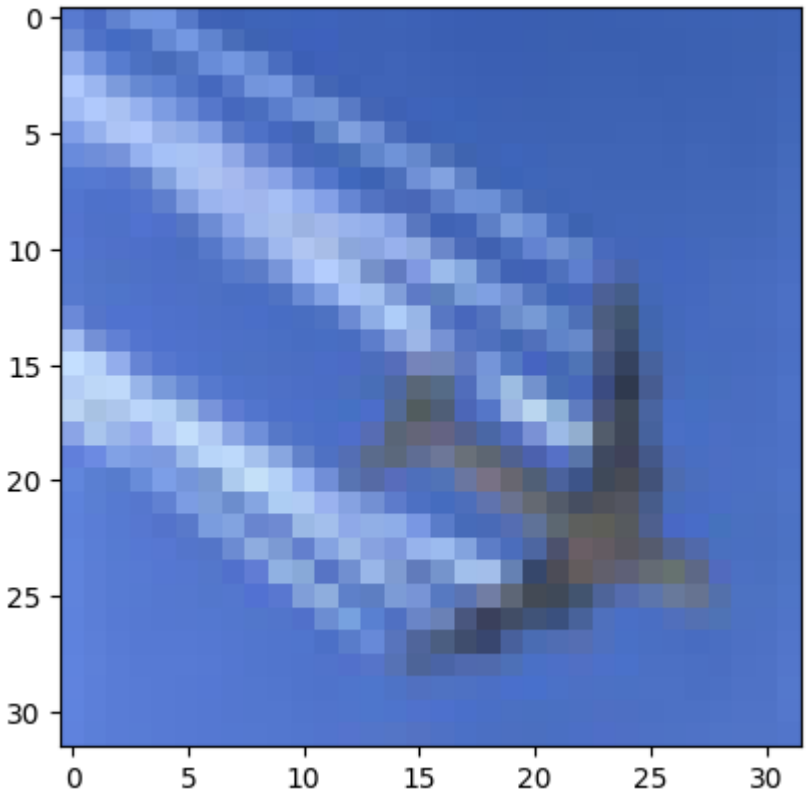
In [ ]:
```
import matplotlib.pyplot as plt

for data, _ in train_dataset.take(1):
    for i in range(9):
        plt.imshow(data[i].numpy().astype('uint8'))
        plt.show()
    break
```

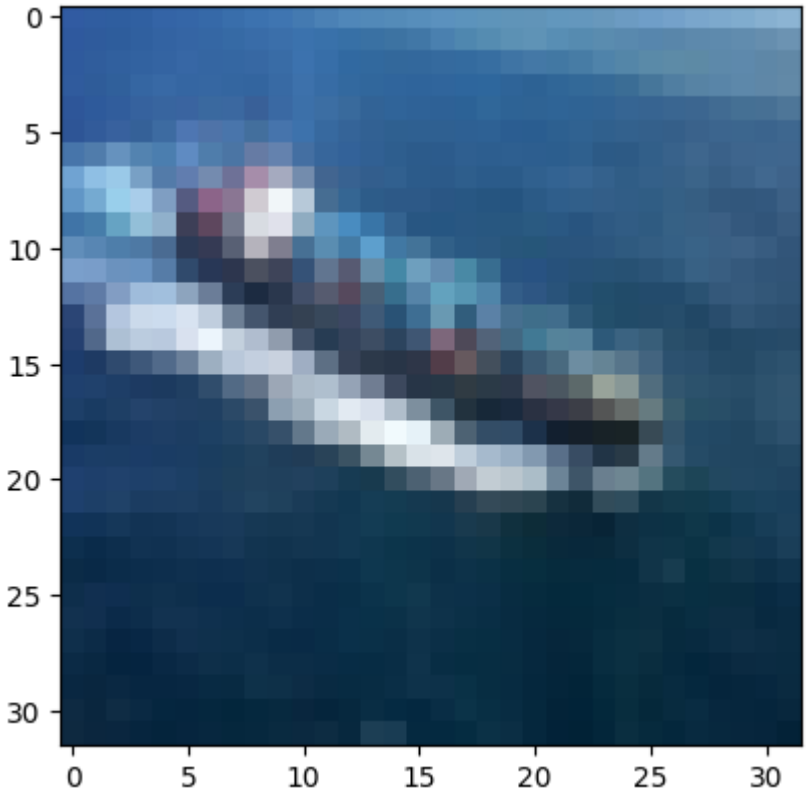


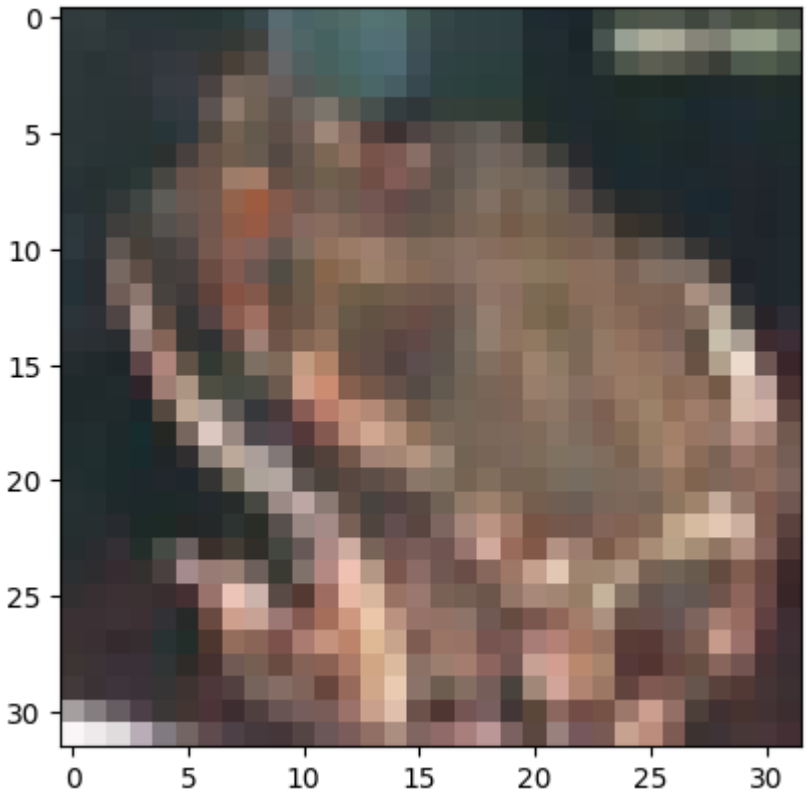

0
5
10
15
20
25
30
0
5
10
15
20
25
30

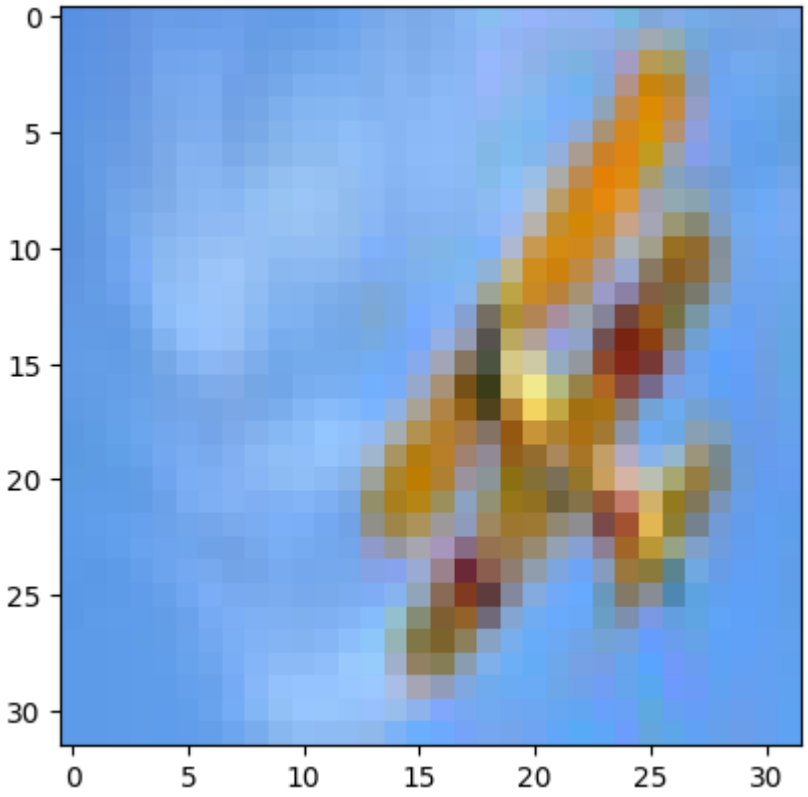

0
5
10
15
20
25
30
0
5
10
15
20
25
30

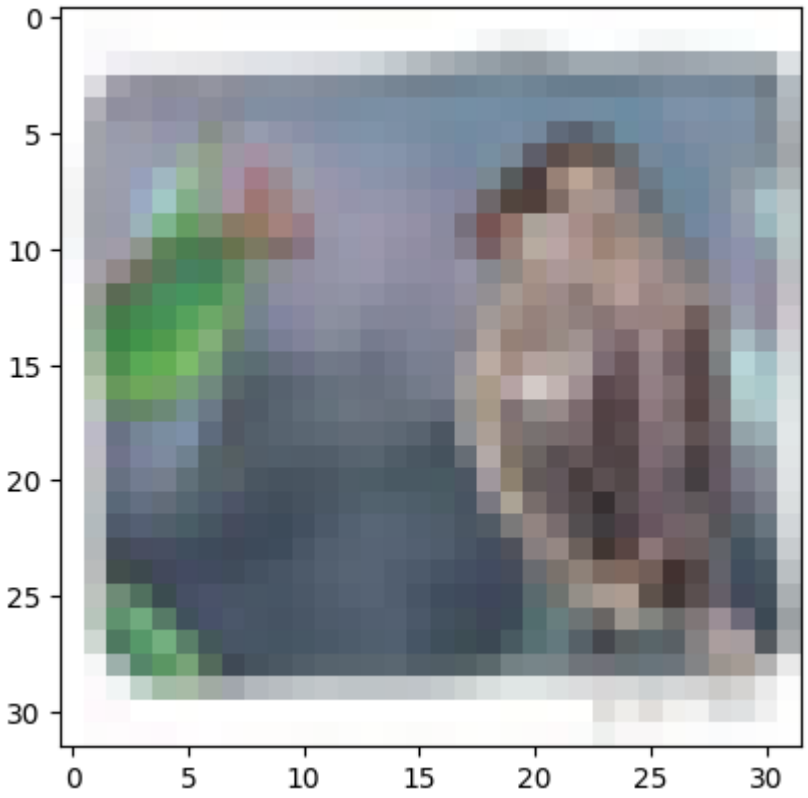

0
5
10
15
20
25
30
0
5
10
15
20
25
30


0
5
10
15
20
25
30
0
5
10
15
20
25
30


0
5
10
15
20
25
30
0
5
10
15
20
25
30


0
5
10
15
20
25
30
0
5
10
15
20
25
30





MODEL ARCHITECTURE

Build a Convolutional Neural Network (CNN) model.

Architecture:

Input -> Conv2D - BN-> Conv2D - BN-> MaxPooling2D -> Conv2D - BN-> Conv2D - BN-> MaxPooling2D -> Flatten -> Dense -> Dropout -> Dense -> Dropout -> Output

1. **Input Layer**
    - The input layer expects images of size 32x32 pixels with 3 color channels (RGB).
    - No Data augmentation is applied.
    - The Rescaling layer, rescales the pixel values from the range [0, 255] to [0, 1].
2. **Convolutional Layers**
    - The model consists of 4 convolutional layers with 32, 64, 128, and 128 filters respectively.
3. **Max Pooling Layers**
    - Max pooling layers are used after each group of 2 convolutional layer to reduce the spatial dimensions of the feature maps.
    - A pooling size of 2x2 is used.
4. **Fully connected layer**
    - A dense layer with 512 units and ReLU activation function.
5. **Output Layer**
    - The output layer consists of 10 units (one for each class) with a softmax activation function.
    - The softmax function outputs the probability distribution over the classes.

Overfitting measures

- Dropout layers are used after each Convolutional and Dense layer to prevent overfitting.
- Kernel Regularization is used to prevent overfitting.

Batch Normalization

- Batch normalization is used after each Convolutional layer to normalize the activations of the previous layer at each batch.
- This helps to stabilize and speed up the training process.

Weight Initialization

- For the `ReLU` activation function, the `he_normal` initializer used to initialize the weights is considered a good weight initialization for ReLU activation functions.
- For the output layer using the `Softmax` activation function, the `glorot_uniform` initializer is used for the same reason.

In [ ]:
```
from tensorflow import keras
from keras import layers, regularizers

inputs = keras.Input(shape=(IMG_SIZE, IMG_SIZE, 3))

x = layers.Rescaling(1./255)(inputs) # Normalize the pixel values to be b
```

```
## First Convolutional Block
x = layers.Conv2D(filters=32, kernel_size=3, kernel_initializer='he_norma
x = layers.BatchNormalization()(x) # Standardize the inputs to the next l
x = layers.Conv2D(filters=64, kernel_size=3, kernel_initializer='he_norma
x = layers.BatchNormalization()(x)

# First Block - Max Pooling and Dropout
x = layers.MaxPooling2D(pool_size=2)(x)
x = layers.Dropout(0.3)(x) # Drops 30% of the neurons randomly during tra

# Second Convolutional Block
x = layers.Conv2D(filters=128, kernel_size=3, kernel_initializer='he_norm
x = layers.BatchNormalization()(x)
x = layers.Conv2D(filters=128, kernel_size=3, kernel_initializer='he_norm
x = layers.BatchNormalization()(x)

# Second Block - Max Pooling and Dropout
x = layers.MaxPooling2D(pool_size=2)(x)
x = layers.Dropout(0.3)(x)

x = layers.Flatten()(x)
x = layers.Dense(512, kernel_initializer='he_normal', activation="relu")(
x = layers.Dropout(0.5)(x)

outputs = layers.Dense(10,  kernel_initializer='glorot_uniform', activati

model = keras.Model(inputs=inputs, outputs=outputs)
model.summary()
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 32, 32, 3)]       0

 rescaling (Rescaling)       (None, 32, 32, 3)         0

 conv2d (Conv2D)             (None, 30, 30, 32)        896

 batch_normalization (Batch  (None, 30, 30, 32)        128
 Normalization)

 conv2d_1 (Conv2D)           (None, 28, 28, 64)        18496

 batch_normalization_1 (Bat  (None, 28, 28, 64)        256
 chNormalization)

 max_pooling2d (MaxPooling2  (None, 14, 14, 64)        0
 D)

 dropout (Dropout)           (None, 14, 14, 64)        0

 conv2d_2 (Conv2D)           (None, 12, 12, 128)       73856

 batch_normalization_2 (Bat  (None, 12, 12, 128)       512
 chNormalization)

 conv2d_3 (Conv2D)           (None, 10, 10, 128)       147584

 batch_normalization_3 (Bat  (None, 10, 10, 128)       512
 chNormalization)

_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 32, 32, 3)]       0

 rescaling (Rescaling)       (None, 32, 32, 3)         0

 conv2d (Conv2D)             (None, 30, 30, 32)        896

 batch_normalization (Batch  (None, 30, 30, 32)        128
 Normalization)

 conv2d_1 (Conv2D)           (None, 28, 28, 64)        18496

 batch_normalization_1 (Bat  (None, 28, 28, 64)        256
 chNormalization)

 max_pooling2d (MaxPooling2  (None, 14, 14, 64)        0
 D)

 dropout (Dropout)           (None, 14, 14, 64)        0

 conv2d_2 (Conv2D)           (None, 12, 12, 128)       73856

 batch_normalization_2 (Bat  (None, 12, 12, 128)       512
 chNormalization)

 conv2d_3 (Conv2D)           (None, 10, 10, 128)       147584
```

```
batch_normalization_3 (Bat  (None, 10, 10, 128)       512
chNormalization)

max_pooling2d_1 (MaxPoolin  (None, 5, 5, 128)         0
g2D)

dropout_1 (Dropout)         (None, 5, 5, 128)         0

flatten (Flatten)           (None, 3200)              0

dense (Dense)               (None, 512)               1638912

dropout_2 (Dropout)         (None, 512)               0

dense_1 (Dense)             (None, 10)                5130

=================================================================
Total params: 1886282 (7.20 MB)
Trainable params: 1885578 (7.19 MB)
Non-trainable params: 704 (2.75 KB)
_________________________________________________________________
```

Compile Model

Loss function:

We use the *Categorical Crossentropy* loss function because it is a `multi-class classification` problem.

Optimizer: Adam

We use the *Adam* optimizer because it is one of the best and most popular optimizers.

In [ ]:
```
model.compile(
    loss='categorical_crossentropy',
    optimizer='adam',
    metrics=['acc'])
```

Train Model

Train the model with Early stopping, Model checkpoint, and Learning rate reduction callbacks.

In [ ]:
```
from keras.callbacks import EarlyStopping, ModelCheckpoint, ReduceLROnPla

learning_rate_reduction = ReduceLROnPlateau(
    monitor='val_acc',
    patience=3,
    verbose=1,
    factor=0.5,
    min_lr=1e-5)

early_stop = EarlyStopping(monitor='val_acc',
                           patience=4,
                           restore_best_weights=True)
model_checkpoint = ModelCheckpoint('models/S01/checkpoints/S01-cp.h5', sa
```

```
history = model.fit(
    train_dataset,
    epochs=100,
    validation_data=validation_dataset,
    callbacks=[early_stop, model_checkpoint, learning_rate_reduction])
```

```
Epoch 1/100
2024-06-22 20:36:05.100481: E tensorflow/core/grappler/optimizers/meta_opt
imizer.cc:961] layout failed: INVALID_ARGUMENT: Size of values 0 does not
match size of permutation 4 @ fanin shape inmodel/dropout/dropout/SelectV
2-2-TransposeNHWCToNCHW-LayoutOptimizer
2024-06-22 20:36:05.299764: I external/local_xla/xla/stream_executor/cuda/
cuda_dnn.cc:454] Loaded cuDNN version 8904
2024-06-22 20:36:07.103978: I external/local_xla/xla/service/service.cc:16
8] XLA service 0x7650e899c7b0 initialized for platform CUDA (this does not
guarantee that XLA will be used). Devices:
2024-06-22 20:36:07.104003: I external/local_xla/xla/service/service.cc:17
6]   StreamExecutor device (0): NVIDIA GeForce GTX 1060 6GB, Compute Capab
ility 6.1
2024-06-22 20:36:07.120937: I tensorflow/compiler/mlir/tensorflow/utils/du
mp_mlir_util.cc:269] disabling MLIR crash reproducer, set env var `MLIR_CR
ASH_REPRODUCER_DIRECTORY` to enable.
WARNING: All log messages before absl::InitializeLog() is called are writt
en to STDERR
I0000 00:00:1719084967.198791  115543 device_compiler.h:186] Compiled clus
ter using XLA!  This line is logged at most once for the lifetime of the p
rocess.
628/628 [==============================] - 18s 20ms/step - loss: 1.7564 -
acc: 0.4143 - val_loss: 1.3404 - val_acc: 0.5566 - lr: 0.0010
Epoch 2/100
  5/628 [..............................] - ETA: 10s - loss: 1.4590 - acc:
0.5094
/home/pws/miniconda3/envs/tensorflow/lib/python3.11/site-packages/keras/sr
c/engine/training.py:3103: UserWarning: You are saving your model as an HD
F5 file via `model.save()`. This file format is considered legacy. We reco
mmend using instead the native Keras format, e.g. `model.save('my_model.ke
ras')`.
  saving_api.save_model(
```

```
628/628 [==============================] - 11s 18ms/step - loss: 1.3207 -
acc: 0.5579 - val_loss: 1.1168 - val_acc: 0.6332 - lr: 0.0010
Epoch 3/100
628/628 [==============================] - 11s 18ms/step - loss: 1.1330 -
acc: 0.6326 - val_loss: 0.9400 - val_acc: 0.6950 - lr: 0.0010
Epoch 4/100
628/628 [==============================] - 11s 17ms/step - loss: 1.0223 -
acc: 0.6744 - val_loss: 0.9063 - val_acc: 0.7104 - lr: 0.0010
Epoch 5/100
628/628 [==============================] - 10s 16ms/step - loss: 0.9399 -
acc: 0.7042 - val_loss: 0.8627 - val_acc: 0.7297 - lr: 0.0010
Epoch 6/100
628/628 [==============================] - 10s 16ms/step - loss: 0.8751 -
acc: 0.7280 - val_loss: 0.8379 - val_acc: 0.7418 - lr: 0.0010
Epoch 7/100
628/628 [==============================] - 10s 16ms/step - loss: 0.8257 -
acc: 0.7502 - val_loss: 0.8385 - val_acc: 0.7491 - lr: 0.0010
Epoch 8/100
628/628 [==============================] - 11s 17ms/step - loss: 0.7850 -
acc: 0.7640 - val_loss: 0.7792 - val_acc: 0.7712 - lr: 0.0010
Epoch 9/100
628/628 [==============================] - 10s 17ms/step - loss: 0.7445 -
acc: 0.7783 - val_loss: 0.8016 - val_acc: 0.7710 - lr: 0.0010
Epoch 10/100
628/628 [==============================] - 11s 18ms/step - loss: 0.7116 -
acc: 0.7921 - val_loss: 0.8480 - val_acc: 0.7663 - lr: 0.0010
Epoch 11/100
628/628 [==============================] - 11s 18ms/step - loss: 0.6778 -
acc: 0.8059 - val_loss: 0.7997 - val_acc: 0.7787 - lr: 0.0010
Epoch 12/100
628/628 [==============================] - 11s 17ms/step - loss: 0.6533 -
acc: 0.8129 - val_loss: 0.7341 - val_acc: 0.7942 - lr: 0.0010
Epoch 13/100
628/628 [==============================] - 11s 17ms/step - loss: 0.6239 -
acc: 0.8261 - val_loss: 0.7874 - val_acc: 0.7872 - lr: 0.0010
Epoch 14/100
628/628 [==============================] - 11s 17ms/step - loss: 0.6032 -
acc: 0.8354 - val_loss: 0.7908 - val_acc: 0.7938 - lr: 0.0010
Epoch 15/100
628/628 [==============================] - ETA: 0s - loss: 0.5908 - acc:
0.8424
Epoch 15: ReduceLROnPlateau reducing learning rate to 0.000500000023748725
7.
628/628 [==============================] - 11s 17ms/step - loss: 0.5908 -
acc: 0.8424 - val_loss: 0.7948 - val_acc: 0.7876 - lr: 0.0010
Epoch 16/100
628/628 [==============================] - 11s 17ms/step - loss: 0.4904 -
acc: 0.8759 - val_loss: 0.7311 - val_acc: 0.8118 - lr: 5.0000e-04
Epoch 17/100
628/628 [==============================] - 11s 17ms/step - loss: 0.4475 -
acc: 0.8866 - val_loss: 0.7362 - val_acc: 0.8125 - lr: 5.0000e-04
Epoch 18/100
628/628 [==============================] - 11s 17ms/step - loss: 0.4275 -
acc: 0.8941 - val_loss: 0.6913 - val_acc: 0.8270 - lr: 5.0000e-04
Epoch 19/100
628/628 [==============================] - 11s 17ms/step - loss: 0.4050 -
acc: 0.9021 - val_loss: 0.7263 - val_acc: 0.8260 - lr: 5.0000e-04
Epoch 20/100
628/628 [==============================] - 11s 17ms/step - loss: 0.3869 -
acc: 0.9076 - val_loss: 0.7334 - val_acc: 0.8201 - lr: 5.0000e-04
```

```
Epoch 21/100
628/628 [==============================] - 11s 17ms/step - loss: 0.3798 -
acc: 0.9110 - val_loss: 0.7047 - val_acc: 0.8307 - lr: 5.0000e-04
Epoch 22/100
628/628 [==============================] - 11s 17ms/step - loss: 0.3651 -
acc: 0.9157 - val_loss: 0.7155 - val_acc: 0.8291 - lr: 5.0000e-04
Epoch 23/100
628/628 [==============================] - 11s 17ms/step - loss: 0.3553 -
acc: 0.9171 - val_loss: 0.7505 - val_acc: 0.8218 - lr: 5.0000e-04
Epoch 24/100
627/628 [==============================>.] - ETA: 0s - loss: 0.3476 - acc:
0.9218
Epoch 24: ReduceLROnPlateau reducing learning rate to 0.000250000011874362
8.
628/628 [==============================] - 11s 17ms/step - loss: 0.3476 -
acc: 0.9218 - val_loss: 0.7527 - val_acc: 0.8241 - lr: 5.0000e-04
Epoch 25/100
628/628 [==============================] - 11s 17ms/step - loss: 0.3059 -
acc: 0.9358 - val_loss: 0.7303 - val_acc: 0.8360 - lr: 2.5000e-04
Epoch 26/100
628/628 [==============================] - 11s 17ms/step - loss: 0.2893 -
acc: 0.9404 - val_loss: 0.7542 - val_acc: 0.8310 - lr: 2.5000e-04
Epoch 27/100
628/628 [==============================] - 11s 17ms/step - loss: 0.2786 -
acc: 0.9432 - val_loss: 0.7600 - val_acc: 0.8310 - lr: 2.5000e-04
Epoch 28/100
628/628 [==============================] - ETA: 0s - loss: 0.2769 - acc:
0.9442
Epoch 28: ReduceLROnPlateau reducing learning rate to 0.000125000005937181
4.
628/628 [==============================] - 11s 17ms/step - loss: 0.2769 -
acc: 0.9442 - val_loss: 0.7239 - val_acc: 0.8331 - lr: 2.5000e-04
Epoch 29/100
628/628 [==============================] - 11s 17ms/step - loss: 0.2510 -
acc: 0.9524 - val_loss: 0.7275 - val_acc: 0.8378 - lr: 1.2500e-04
Epoch 30/100
628/628 [==============================] - 11s 17ms/step - loss: 0.2450 -
acc: 0.9549 - val_loss: 0.7159 - val_acc: 0.8394 - lr: 1.2500e-04
Epoch 31/100
628/628 [==============================] - 11s 17ms/step - loss: 0.2417 -
acc: 0.9563 - val_loss: 0.7460 - val_acc: 0.8390 - lr: 1.2500e-04
Epoch 32/100
628/628 [==============================] - 11s 17ms/step - loss: 0.2362 -
acc: 0.9571 - val_loss: 0.7418 - val_acc: 0.8433 - lr: 1.2500e-04
Epoch 33/100
628/628 [==============================] - 11s 17ms/step - loss: 0.2305 -
acc: 0.9577 - val_loss: 0.7368 - val_acc: 0.8411 - lr: 1.2500e-04
Epoch 34/100
628/628 [==============================] - 11s 17ms/step - loss: 0.2273 -
acc: 0.9595 - val_loss: 0.7566 - val_acc: 0.8386 - lr: 1.2500e-04
Epoch 35/100
627/628 [==============================>.] - ETA: 0s - loss: 0.2237 - acc:
0.9609
Epoch 35: ReduceLROnPlateau reducing learning rate to 6.25000029685907e-0
5.
628/628 [==============================] - 11s 17ms/step - loss: 0.2237 -
acc: 0.9609 - val_loss: 0.7342 - val_acc: 0.8396 - lr: 1.2500e-04
Epoch 36/100
628/628 [==============================] - 11s 17ms/step - loss: 0.2136 -
acc: 0.9630 - val_loss: 0.7524 - val_acc: 0.8417 - lr: 6.2500e-05
```

Save Model

In [ ]:
```
keras.models.save_model(model, 'models/S01/S01-model.h5')
```

```
/tmp/ipykernel_104651/2467668447.py:1: UserWarning: You are saving your mo
del as an HDF5 file via `model.save()`. This file format is considered leg
acy. We recommend using instead the native Keras format, e.g. `model.save
('my_model.keras')`.
  keras.models.save_model(model, 'models/S01/S01-model.h5')
```

Load Model

In [ ]:
```
from tensorflow import keras
keras.models.load_model('models/S01/S01-model.h5')
```

Out[ ]:
```
<keras.src.engine.functional.Functional at 0x7651980a9f90>
```

EVALUATION

Evaluate the model on the validation dataset.

In [ ]:
```
val_loss, val_acc = model.evaluate(validation_dataset)
print('val_acc:', val_acc)
```

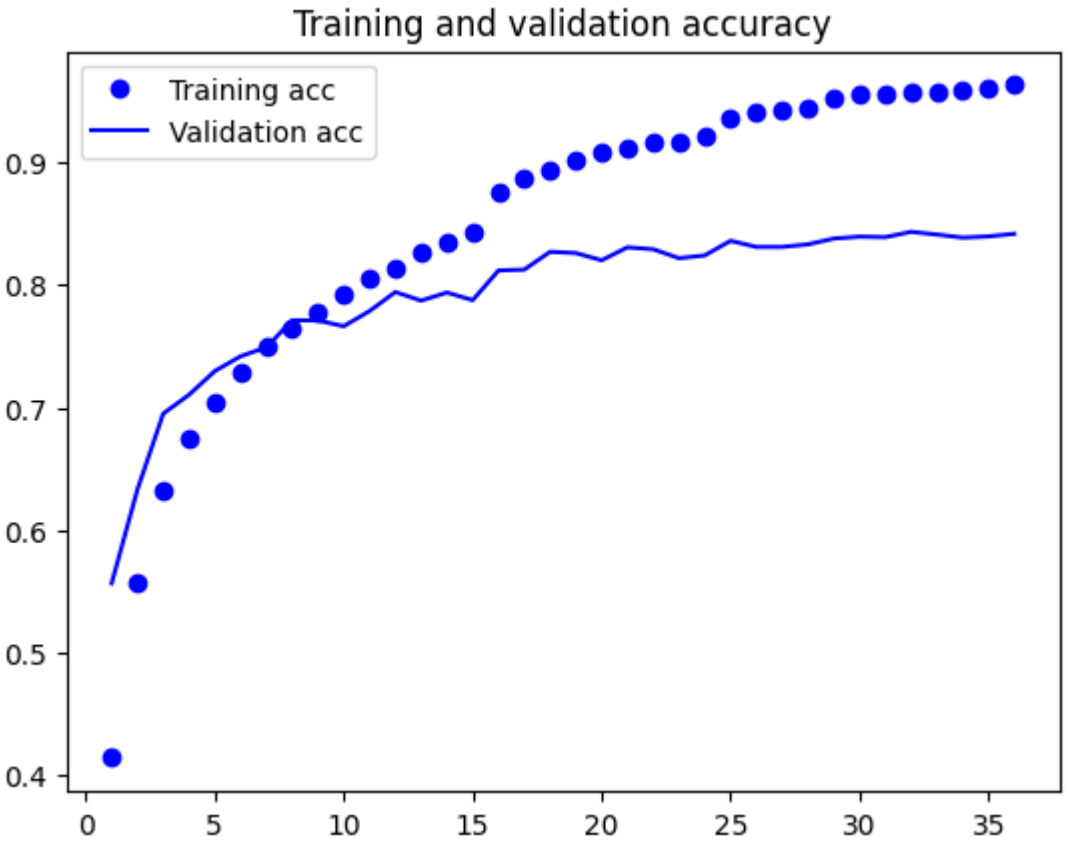
```
157/157 [==============================] - 1s 5ms/step - loss: 0.7418 - ac
c: 0.8433
val_acc: 0.8432999849319458
```

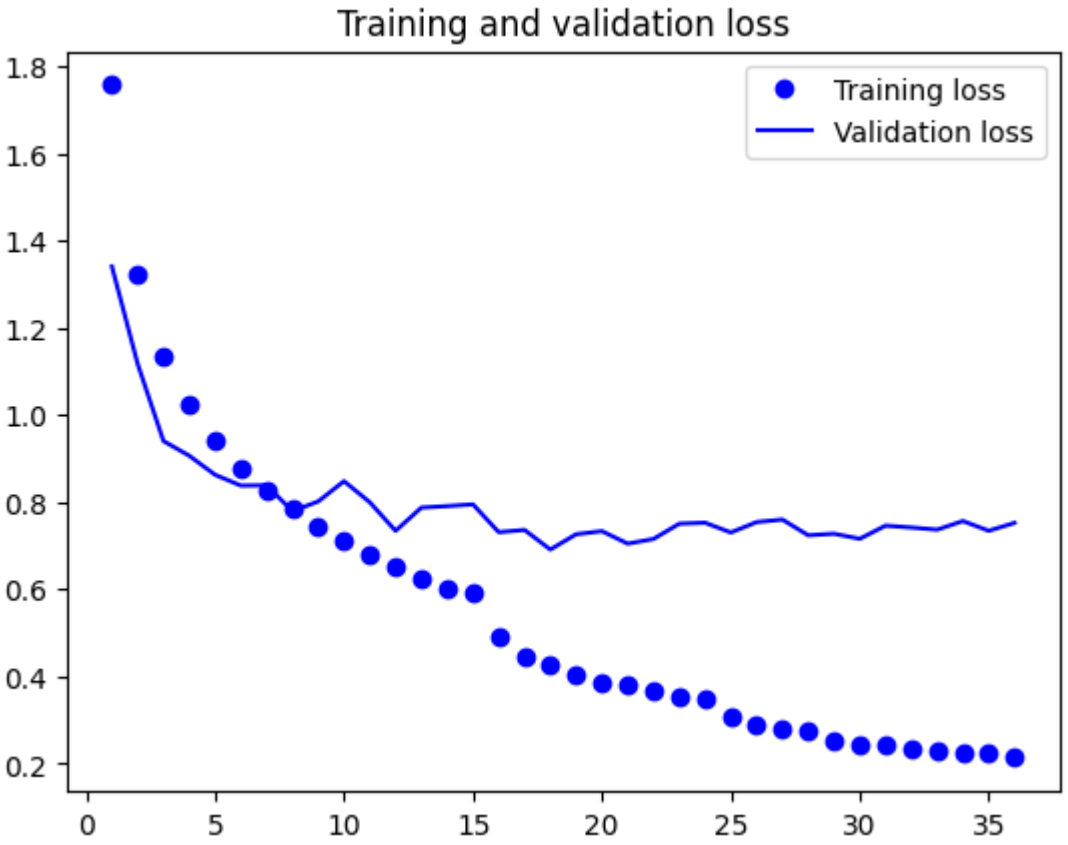
Training and Validation Curves

Plot the training and validation accuracy and loss curves.

In [ ]:
```
import matplotlib.pyplot as plt

# Extract the history from the training process
acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(acc) + 1)

# Plot the training and validation accuracy
plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.legend()

# Plot the training and validation loss
plt.figure()
plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
```

```
plt.legend()
plt.show()
```





Confusion Matrix

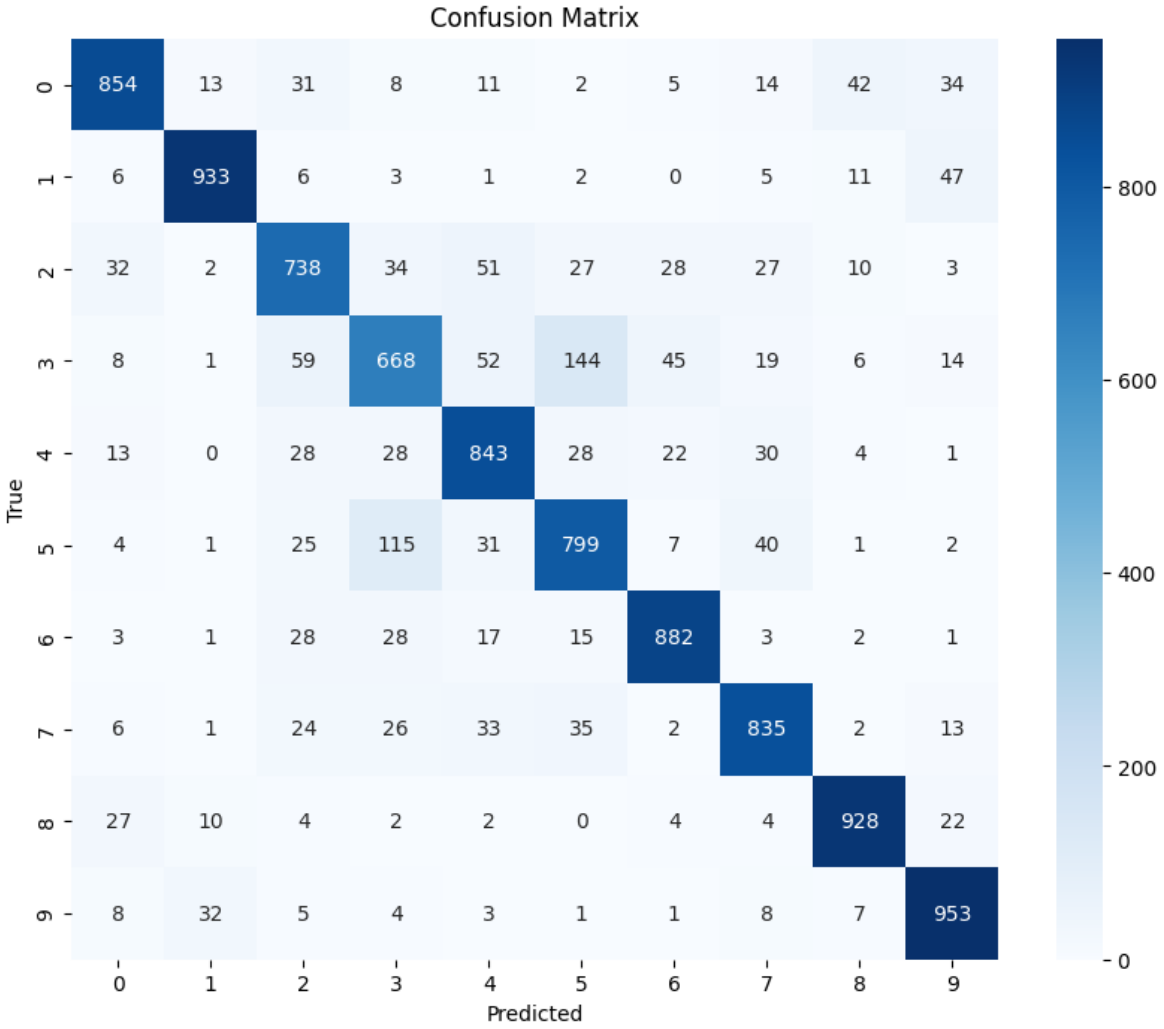
In [ ]:
```
import numpy as np
```

```
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix

y_true = []
y_pred = []

for features, labels in validation_dataset:
    predictions = model.predict(features)
    y_true.extend(np.argmax(labels.numpy(), axis=1))
    y_pred.extend(np.argmax(predictions, axis=1))

y_true = np.array(y_true)
y_pred = np.array(y_pred)

cm = confusion_matrix(y_true, y_pred)

# Plot the confusion matrix
plt.figure(figsize=(10, 8))
sns.heatmap(cm, annot=True, cmap='Blues', fmt='g')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix')
plt.show()
```

```
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
```

```
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
```

```
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
2/2 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 107ms/step
```



In [ ]:
```
print("Confusion Matrix:")
print(cm)
```

```
Confusion Matrix:
[[854  13  31   8  11   2   5  14  42  34]
 [  6 933   6   3   1   2   0   5  11  47]
 [ 32   2 738  34  51  27  28  27  10   3]
 [  8   1  59 668  52 144  45  19   6  14]
 [ 13   0  28  28 843  28  22  30   4   1]
 [  4   1  25 115  31 799   7  40   1   2]
 [  3   1  28  28  17  15 882   3   2   1]
 [  6   1  24  26  33  35   2 835   2  13]
 [ 27  10   4   2   2   0   4   4 928  22]
 [  8  32   5   4   3   1   1   8   7 953]]
```

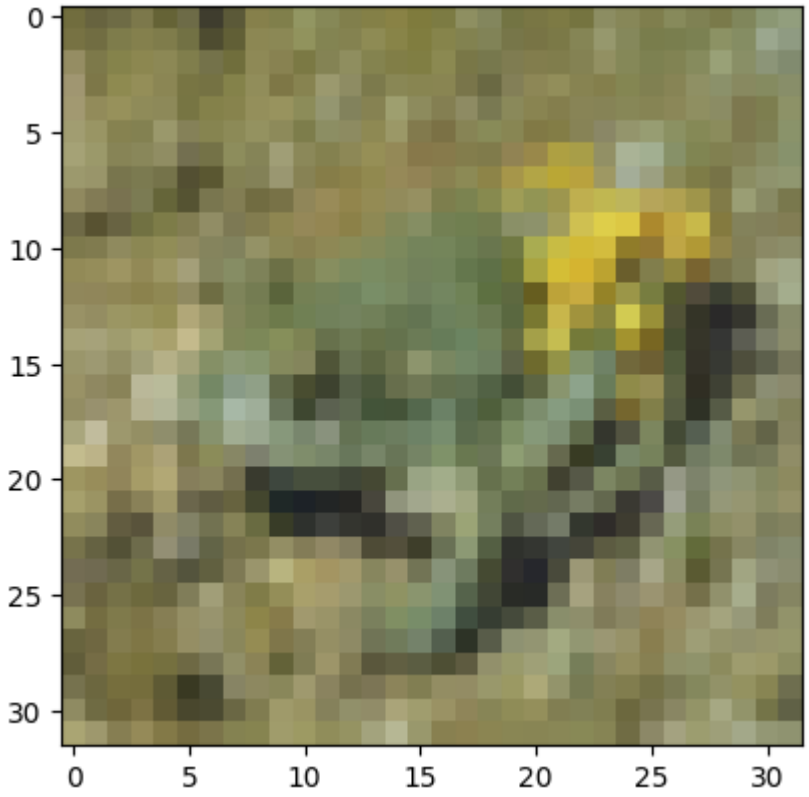
In [ ]:
```
from sklearn.metrics import classification_report

report = classification_report(y_true, y_pred, target_names=class_names)
print(report)
```

	precision	recall	f1-score	support
airplane	0.89	0.84	0.86	1014
automobile	0.94	0.92	0.93	1014
bird	0.78	0.78	0.78	952
cat	0.73	0.66	0.69	1016
deer	0.81	0.85	0.83	997
dog	0.76	0.78	0.77	1025
frog	0.89	0.90	0.89	980
horse	0.85	0.85	0.85	977
ship	0.92	0.93	0.92	1003
truck	0.87	0.93	0.90	1022
accuracy			0.84	10000
macro avg	0.84	0.84	0.84	10000
weighted avg	0.84	0.84	0.84	10000

Predictions

Predict and visualize the results for a sample image.

In [ ]:
```
import tensorflow as tf
import matplotlib.pyplot as plt
from keras.preprocessing import image

# Load an image
img = tf.keras.preprocessing.image.load_img(train_dirs[0] + '/006_frog/al
# img = tf.keras.preprocessing.image.load_img(train_dirs[0] + '/000_airpl

# Preprocess the image
img_array = image.img_to_array(img)
img_array = tf.expand_dims(img_array, 0)

plt.imshow(img)
plt.show()

print(img_array.shape)
result = model.predict(img_array)
print("Result: ", result.round())
```




```
(1, 32, 32, 3)
1/1 [==============================] - 0s 222ms/step
Result:  [[0. 0. 0. 0. 0. 0. 1. 0. 0. 0.]]
```