A1_Part3_solution

September 10, 2023

0.1 Part 3: Convolutional Neural Networks and Image Classification

[Total marks for this part: 40 points]

This part of the asssignment is designed to assess your knowledge and coding skill with Tensorflow as well as hands-on experience with training Convolutional Neural Network (CNN).

The dataset we use for this part is the STL10 dataset which consists of 5,000 training images of airplane, bird, car, cat, deer, dog, horse, monkey, ship, truck; each of which has 500 images. You can download the dataset at download here and then decompress to the folder `datasets\Animals` in your assignment folder.**

Your task is to build a CNN model using *TF 2.x* to classify these animals. You're provided with the module models.py, which you can find in the assignment folder, with some of the following classes:

1. `DatasetManager`: Support with loading and spliting the dataset into the train-val-test sets. It also supports generating next batches for training.
2. `BaseImageClassifier`: A base class image classfication, which is basically a CNN model.

Note: You may need to install the package `imutils` if you have not installed yet

Firstly, we need to run the following cells to load required packages.

[ ]:
```
%load_ext autoreload
%autoreload 2
```

[ ]:
```
import os
import tensorflow as tf
import matplotlib.pyplot as plt
plt.style.use('ggplot')
%matplotlib inline
from A1_S2_2023 import DatasetManager, BaseImageClassifier
```

Note that the class `DatasetManager` has attributes related to *the training, validation, and testing sets*. You can use them in training your developped models in the sequel.

[ ]:
```
dataset_name = 'stl10'
# Choose path to store dataset
data_dir = '{}/tensorflow_datasets'.format(os.path.expanduser('~'))

data_manager = DatasetManager(dataset_name, data_dir)
data_manager.load_dataset()
```

```
data_manager.preprocess_dataset()
data_manager.show_examples()
```

```
Downloading and preparing dataset 2.46 GiB (download: 2.46 GiB, generated: 1.86
GiB, total: 4.32 GiB) to /root/tensorflow_datasets/stl10/1.0.0…

Dl Completed…: 0 url [00:00, ? url/s]

Dl Size…: 0 MiB [00:00, ? MiB/s]

Extraction completed…: 0 file [00:00, ? file/s]

Generating splits…:   0%|          | 0/3 [00:00<?, ? splits/s]

Generating train examples…:   0%|          | 0/5000 [00:00<?, ? examples/s]

Shuffling /root/tensorflow_datasets/stl10/1.0.0.incompleteHZ8LUJ/stl10-train.
↪tfrecord*…:   0%|          | 0/…

Generating test examples…:   0%|          | 0/8000 [00:00<?, ? examples/s]

Shuffling /root/tensorflow_datasets/stl10/1.0.0.incompleteHZ8LUJ/stl10-test.
↪tfrecord*…:   0%|          | 0/8…

Generating unlabelled examples…:   0%|          | 0/100000 [00:00<?, ? examples/
↪s]

Shuffling /root/tensorflow_datasets/stl10/1.0.0.incompleteHZ8LUJ/
↪stl10-unlabelled.tfrecord*…:   0%|          …

Dataset stl10 downloaded and prepared to /root/tensorflow_datasets/stl10/1.0.0.
Subsequent calls will reuse this data.
```

[ ]:

```
# Choose a random example
import random
num_examples = tf.data.experimental.cardinality(data_manager.ds_train).numpy()
random_index = random.randint(0, num_examples - 1)
example = next(iter(data_manager.ds_train.skip(random_index).take(1)))[0]

# Print the shape and value of the image
print("Image shape:", example.shape)
print("Image value range:", example.numpy().min(), "to", example.numpy().max())
```

```
Image shape: (32, 32, 3)
Image value range: 0.09803922 to 0.9647059
```

[ ]:
```
# Check the number of examples in each dataset
print(tf.data.experimental.cardinality(data_manager.ds_train))
print(tf.data.experimental.cardinality(data_manager.ds_val))
print(tf.data.experimental.cardinality(data_manager.ds_test))
```

```
tf.Tensor(4500, shape=(), dtype=int64)
tf.Tensor(500, shape=(), dtype=int64)
tf.Tensor(8000, shape=(), dtype=int64)
```

We now use **BaseImageClassifier** built in the **A1_S2_2023.py** file which serves as a basic baseline to start the investigation. Follow the following steps to realize how to run a model and know the built-in methods associated with.

[ ]:
```
network1 = BaseImageClassifier(name='network1',
                       num_classes=10,
                       optimizer='sgd',
                       batch_size=128,
                       num_epochs=20,
                       learning_rate=0.001)
```

We first initialize a default model from the DefaultModel class. Basically, we can define the relevant parameters of training a model including `num_classes`, `optimizer`, `learning_rate`, `batch_size`, and `num_epochs`.

The method `build_cnn()` assists us in building your convolutional neural network. You can view the code (in the **A1_S2_2023.py** file) of the model behind a default model to realize how simple it is. Additionally, the method `summary()` shows the architecture of a model.

[ ]:
```
network1.build_cnn()
network1.summary()
```

```
Model: "sequential_1"

_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 32, 32, 32)        896

 conv2d_1 (Conv2D)           (None, 32, 32, 32)        9248

 average_pooling2d (AverageP  (None, 16, 16, 32)       0
 ooling2D)

 conv2d_2 (Conv2D)           (None, 16, 16, 64)        18496

 conv2d_3 (Conv2D)           (None, 16, 16, 64)        36928

 average_pooling2d_1 (Averag  (None, 8, 8, 64)         0
 ePooling2D)
```

```
 flatten (Flatten)           (None, 4096)              0

 dense (Dense)               (None, 10)                40970

=================================================================
Total params: 106,538
Trainable params: 106,538
Non-trainable params: 0
_________________________________________________________________
None
```

[ ]:
```
x_train_batch = network1.optimize_data_pipeline(data_manager.ds_train,␣
 ↪batch_size=32)
x_val_batch = network1.optimize_data_pipeline(data_manager.ds_val,␣
 ↪batch_size=32)
network1.fit(x_train_batch, x_val_batch, num_epochs=20)
```

```
Epoch 1/20
141/141 [==============================] - 16s 31ms/step - loss: 2.2981 -
accuracy: 0.1222 - val_loss: 2.2889 - val_accuracy: 0.1500
Epoch 2/20
141/141 [==============================] - 1s 5ms/step - loss: 2.2765 -
accuracy: 0.1427 - val_loss: 2.2474 - val_accuracy: 0.1940
Epoch 3/20
141/141 [==============================] - 1s 5ms/step - loss: 2.1806 -
accuracy: 0.1969 - val_loss: 2.1057 - val_accuracy: 0.2500
Epoch 4/20
141/141 [==============================] - 1s 5ms/step - loss: 2.0548 -
accuracy: 0.2629 - val_loss: 1.9850 - val_accuracy: 0.2960
Epoch 5/20
141/141 [==============================] - 1s 4ms/step - loss: 1.9539 -
accuracy: 0.3029 - val_loss: 1.9003 - val_accuracy: 0.3360
Epoch 6/20
141/141 [==============================] - 1s 4ms/step - loss: 1.8941 -
accuracy: 0.3187 - val_loss: 1.8511 - val_accuracy: 0.3460
Epoch 7/20
141/141 [==============================] - 1s 5ms/step - loss: 1.8360 -
accuracy: 0.3376 - val_loss: 1.8105 - val_accuracy: 0.3560
Epoch 8/20
141/141 [==============================] - 1s 5ms/step - loss: 1.7824 -
accuracy: 0.3649 - val_loss: 1.7775 - val_accuracy: 0.3620
Epoch 9/20
141/141 [==============================] - 1s 5ms/step - loss: 1.7403 -
accuracy: 0.3756 - val_loss: 1.7436 - val_accuracy: 0.3720
Epoch 10/20
141/141 [==============================] - 1s 4ms/step - loss: 1.7052 -
accuracy: 0.3900 - val_loss: 1.7134 - val_accuracy: 0.3800
Epoch 11/20
```

```
141/141 [==============================] - 1s 6ms/step - loss: 1.6717 -
accuracy: 0.4027 - val_loss: 1.6897 - val_accuracy: 0.3780
Epoch 12/20
141/141 [==============================] - 1s 6ms/step - loss: 1.6402 -
accuracy: 0.4124 - val_loss: 1.6700 - val_accuracy: 0.3880
Epoch 13/20
141/141 [==============================] - 1s 5ms/step - loss: 1.6102 -
accuracy: 0.4251 - val_loss: 1.6527 - val_accuracy: 0.3940
Epoch 14/20
141/141 [==============================] - 1s 5ms/step - loss: 1.5811 -
accuracy: 0.4329 - val_loss: 1.6379 - val_accuracy: 0.4040
Epoch 15/20
141/141 [==============================] - 1s 4ms/step - loss: 1.5540 -
accuracy: 0.4376 - val_loss: 1.6254 - val_accuracy: 0.4260
Epoch 16/20
141/141 [==============================] - 1s 5ms/step - loss: 1.5261 -
accuracy: 0.4478 - val_loss: 1.6139 - val_accuracy: 0.4340
Epoch 17/20
141/141 [==============================] - 1s 4ms/step - loss: 1.4978 -
accuracy: 0.4564 - val_loss: 1.6024 - val_accuracy: 0.4440
Epoch 18/20
141/141 [==============================] - 1s 5ms/step - loss: 1.4693 -
accuracy: 0.4700 - val_loss: 1.5910 - val_accuracy: 0.4540
Epoch 19/20
141/141 [==============================] - 1s 4ms/step - loss: 1.4404 -
accuracy: 0.4800 - val_loss: 1.5783 - val_accuracy: 0.4600
Epoch 20/20
141/141 [==============================] - 1s 5ms/step - loss: 1.4111 -
accuracy: 0.4931 - val_loss: 1.5668 - val_accuracy: 0.4560
```

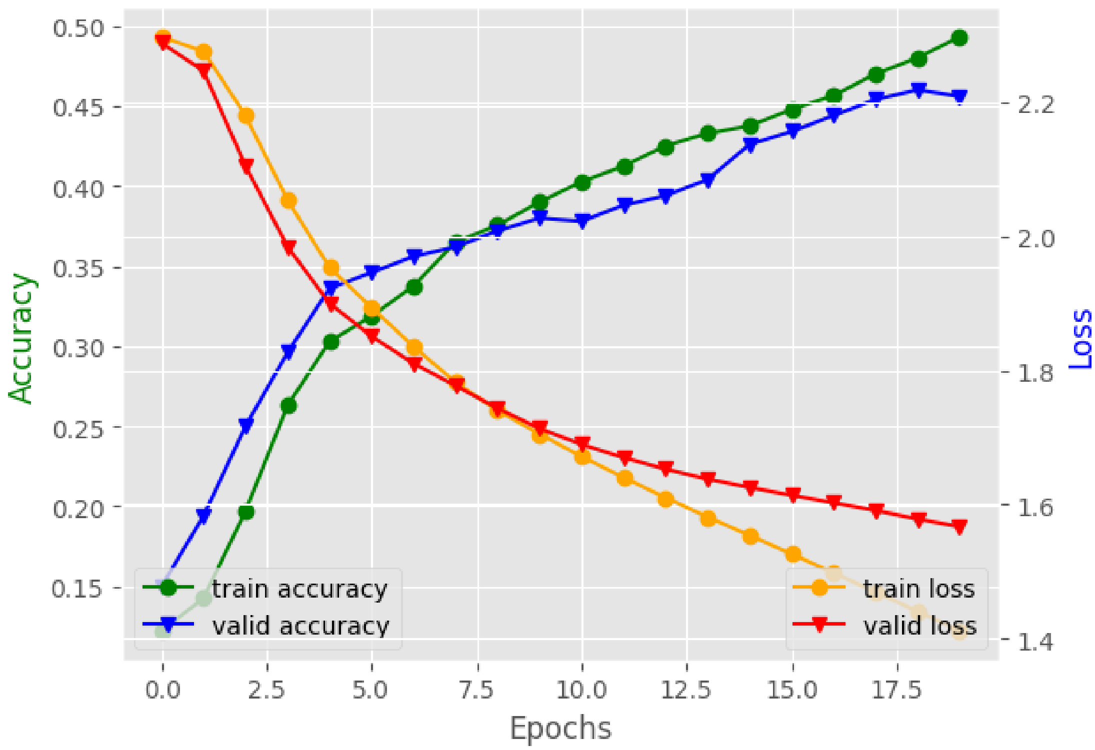
To train a model regarding to the datasets stored in `data_manager`, you can invoke the method `fit()` for which you can specify the batch size and number of epochs for your training.

[ ]:
```
x_test_batch = network1.optimize_data_pipeline(data_manager.ds_test,
    batch_size=32)
network1.compute_accuracy(x_test_batch)
```

```
250/250 [==============================] - 4s 17ms/step - loss: 1.5981 -
accuracy: 0.4289
loss: 1.5980982780456543
accuracy: 0.42887499928474426
```

Below shows how you can inspect the training progress.

[ ]:
```
network1.plot_progress()
```



You can use the method `predict()` to predict labels for data examples in a test set.

[ ]:
```
num_samples = 25
sample_dataset = data_manager.ds_test.take(num_samples)
network1.predict(sample_dataset.batch(num_samples), data_manager.ds_info)
```

```
1/1 [==============================] - 0s 298ms/step
Sample 1: Predicted label - ship
Sample 2: Predicted label - monkey
Sample 3: Predicted label - cat
Sample 4: Predicted label - bird
Sample 5: Predicted label - ship
Sample 6: Predicted label - dog
Sample 7: Predicted label - deer
Sample 8: Predicted label - cat
Sample 9: Predicted label - airplane
Sample 10: Predicted label - deer
Sample 11: Predicted label - deer
Sample 12: Predicted label - ship
Sample 13: Predicted label - horse
Sample 14: Predicted label - ship
Sample 15: Predicted label - deer
Sample 16: Predicted label - cat
```

```
Sample 17: Predicted label - deer
Sample 18: Predicted label - airplane
Sample 19: Predicted label - truck
Sample 20: Predicted label - ship
Sample 21: Predicted label - dog
Sample 22: Predicted label - horse
Sample 23: Predicted label - ship
Sample 24: Predicted label - truck
Sample 25: Predicted label - airplane
```

Finally, the method `plot_prediction()` visualizes the predictions for a test set in which several images are chosen to show the predictions.

[ ]:
```
num_samples = 20
sample_dataset = data_manager.ds_test.take(num_samples)
network1.plot_predictions(sample_dataset, data_manager.ds_info,
    num_samples=num_samples, grid_shape=(4, 5))
```

```
1/1 [==============================] - 0s 79ms/step
```

0.1.1 Question 3.1: Observe the learning curve

After running the above cells to train the default model and observe the learning curve. Report your observation (i.e. did the model learn well? if not, what is the problem? What would you do to improve it?). Write your answer below.

[4 points]

The model behaves not too goood. The accuracy for test dataset is only 0.422.

What could be done to immprove: increase number of epoches, add skip connection, add dropout

For questions 3.2 to 3.9, you'll need to write your own model in a way that makes it easy for you to experiment with different architectures and parameters. The goal is to be able to pass the parameters to initialize a new instance of `YourModel` to build different network architectures with different parameters. Below are descriptions of some parameters which you can find in function `__init__()` for the class `BaseImageClassifier`:

1. `num_blocks`: an integer specifying the number of blocks in our network. Each block has the pattern `[conv, batch norm, activation, conv, batch norm, activation, mean pool, dropout]`. All convolutional layers have filter size $(3, 3)$, strides $(1, 1)$ and 'SAME' padding, and all mean pool layers have strides $(2, 2)$ and 'SAME' padding. The network will consists of a few blocks before applying a linear layer to output the logits for the softmax layer.
2. `feature_maps`: the number of feature maps in the first block of the network. The number of feature_maps will double in each of the following block. To make it convenient for you, we already calculated the number of feature maps for each block for you in line 106
3. `drop_rate`: the keep probability for dropout. Setting `drop_rate` to 0.0 means not using dropout.
4. `batch_norm`: the batch normalization function is used or not. Setting `batch_norm` to `None` means not using batch normalization.
5. The `skip connection` is added to the output of the second `batch norm`. Additionally, your class has a boolean property (i.e., instance variable) named `use_skip`. If `use_skip=True`, the skip connectnion is enable. Otherwise, if `use_skip=False`, the skip connectnion is disable.

Below is the architecture of one block:

Below is the architecture of the entire deep net with `two blocks`:

Here we assume that the first block has `feature_maps = feature_maps[0] = 32`. Note that the initial number of feature maps of the first block is declared in the instance variable `feature_maps` and is multiplied by 2 in each follpwing block.

[ ]:
```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers, models

tf.random.set_seed(3181)
```

0.1.2 Question 3.2: Define your CNN

Write the code of the `YourModel` class here. Note that this class will be inherited from the `BaseImageClassifier` class. You'll only need to re-write the code for the `build_cnn` method in the `YourModel` class from the cell below.

[4 points]

```
[ ]: import tensorflow as tf
from tensorflow.keras.layers import Input, Conv2D, BatchNormalization,␣
 ↪Activation, AveragePooling2D, Flatten, Dropout
from tensorflow.keras.models import Model


class YourModel(BaseImageClassifier):
    def __init__(self,
                 name='network1',
                 width=32, height=32, depth=3,
                 num_blocks=2,
                 feature_maps=32,
                 num_classes=4,
                 drop_rate=0.2,
                 batch_norm=None,
                 is_augmentation=False,
                 activation_func='relu',
                 optimizer='adam',
                 use_skip=True,
                 batch_size=10,
                 num_epochs=20,
                 learning_rate=0.0001,
                 verbose=True):
        super(YourModel, self).__init__(name, width, height, depth, num_blocks,␣
 ↪feature_maps, num_classes, drop_rate, batch_norm, is_augmentation,
                                        activation_func, use_skip, optimizer,␣
 ↪batch_size, num_epochs, learning_rate, verbose)

    def custom_block(self, x, filters):
        # skip connection
        shortcut = x

        # First Conv layer
        x = layers.Conv2D(filters=filters, kernel_size=(3, 3), strides=(1, 1),␣
 ↪padding="same")(x)
        if self.batch_norm:
            x = layers.BatchNormalization()(x)
        x = layers.Activation(self.activation_func)(x)

        # Second Conv layer
```

```
        x = layers.Conv2D(filters=filters, kernel_size=(3, 3), strides=(1, 1),␣
↪padding="same")(x)
        if self.batch_norm:
            x = layers.BatchNormalization()(x)

        # Add the skip connection to the output
        if self.use_skip:
            x = tf.keras.layers.add([shortcut, x])
        x = layers.Activation(self.activation_func)(x)

        # Mean Pooling layer
        x = layers.AveragePooling2D(pool_size=(2, 2), strides=(2, 2),␣
↪padding="same")(x)

        # Dropout
        x = layers.Dropout(rate=self.drop_rate)(x)

        return x

    def build_cnn(self):
      input_tensor = layers.Input(shape=(self.height, self.width, self.depth))
      x = input_tensor

      for current_feature_maps in self.feature_maps:
        # Initial convolution
        x = layers.Conv2D(current_feature_maps, (3, 3), padding='same',␣
↪activation=None)(x)
        if self.batch_norm:
            x = layers.BatchNormalization()(x)
        x = layers.Activation(self.activation_func)(x)

        # Second convolution with skip connection
        shortcut = x
        x = layers.Conv2D(current_feature_maps, (3, 3), padding='same',␣
↪activation=None)(x)
        if self.batch_norm:
            x = layers.BatchNormalization()(x)
        if self.use_skip:
            x = layers.Add()([shortcut, x])
        x = layers.Activation(self.activation_func)(x)

        # Mean Pooling and Dropout
        x = layers.AveragePooling2D(pool_size=(2, 2), padding='same')(x)
        if self.drop_rate > 0:
            x = layers.Dropout(self.drop_rate)(x)

      x = layers.Flatten()(x)
```

```
    x = layers.Dense(self.num_classes, activation='softmax')(x)

    self.model = models.Model(inputs=input_tensor, outputs=x)
    self.model.compile(optimizer=self.optimizer,␣
↪loss='sparse_categorical_crossentropy', metrics=['accuracy'])
```

0.1.3 Question 3.3: Experiment with skip connection

Once writing your own model, you need to compare two cases: (i) *using the skip connection* and (ii) *not using the skip connection*. You should set the instance variable `use_skip` to either `True` or `False`. For your runs, report which case is better and if you confront overfitting in training.

[6 points]

 # WRITE YOUR ANSWER AND OBSERVATION HERE

…..

[ ]:
```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers, models

my_network_skip = YourModel(name='network1',
                     feature_maps=32,
                     num_classes=data_manager.n_classes,
                     num_blocks=3,
                     drop_rate=0.0,
                     batch_norm=True,
                     use_skip=True,
                     optimizer='sgd',
                     learning_rate=0.001)
my_network_skip.build_cnn()
my_network_skip.summary()
```

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

 conv2d_4 (Conv2D)              (None, 32, 32, 32)   896
['input_1[0][0]']

 batch_normalization (BatchNorm  (None, 32, 32, 32)  128
['conv2d_4[0][0]']
 alization)
```

```
 activation (Activation)        (None, 32, 32, 32)   0
['batch_normalization[0][0]']

 conv2d_5 (Conv2D)              (None, 32, 32, 32)   9248
['activation[0][0]']

 batch_normalization_1 (BatchNo  (None, 32, 32, 32)  128
['conv2d_5[0][0]']
 rmalization)

 add (Add)                      (None, 32, 32, 32)   0
['activation[0][0]',
'batch_normalization_1[0][0]']

 activation_1 (Activation)      (None, 32, 32, 32)   0            ['add[0][0]']

 average_pooling2d_2 (AveragePo  (None, 16, 16, 32)  0
['activation_1[0][0]']
 oling2D)

 conv2d_6 (Conv2D)              (None, 16, 16, 64)   18496
['average_pooling2d_2[0][0]']

 batch_normalization_2 (BatchNo  (None, 16, 16, 64)  256
['conv2d_6[0][0]']
 rmalization)

 activation_2 (Activation)      (None, 16, 16, 64)   0
['batch_normalization_2[0][0]']

 conv2d_7 (Conv2D)              (None, 16, 16, 64)   36928
['activation_2[0][0]']

 batch_normalization_3 (BatchNo  (None, 16, 16, 64)  256
['conv2d_7[0][0]']
 rmalization)

 add_1 (Add)                    (None, 16, 16, 64)   0
['activation_2[0][0]',
'batch_normalization_3[0][0]']

 activation_3 (Activation)      (None, 16, 16, 64)   0            ['add_1[0][0]']

 average_pooling2d_3 (AveragePo  (None, 8, 8, 64)    0
['activation_3[0][0]']
 oling2D)

 conv2d_8 (Conv2D)              (None, 8, 8, 128)    73856
```

```
['average_pooling2d_3[0][0]']

 batch_normalization_4 (BatchNo  (None, 8, 8, 128)   512
['conv2d_8[0][0]']
 rmalization)

 activation_4 (Activation)      (None, 8, 8, 128)    0
['batch_normalization_4[0][0]']

 conv2d_9 (Conv2D)              (None, 8, 8, 128)    147584
['activation_4[0][0]']

 batch_normalization_5 (BatchNo  (None, 8, 8, 128)   512
['conv2d_9[0][0]']
 rmalization)

 add_2 (Add)                    (None, 8, 8, 128)    0
['activation_4[0][0]',
'batch_normalization_5[0][0]']

 activation_5 (Activation)      (None, 8, 8, 128)    0            ['add_2[0][0]']

 average_pooling2d_4 (AveragePo  (None, 4, 4, 128)   0
['activation_5[0][0]']
 oling2D)

 flatten_1 (Flatten)            (None, 2048)         0
['average_pooling2d_4[0][0]']

 dense_1 (Dense)                (None, 10)           20490
['flatten_1[0][0]']

================================================================================
==================
Total params: 309,290
Trainable params: 308,394
Non-trainable params: 896
--------------------------------------------------------------------------------
------------------
None
```

[ ]:
```
my_network_skip.fit(x_train_batch, x_val_batch, num_epochs=20)
```

```
Epoch 1/20
141/141 [==============================] - 3s 9ms/step - loss: 1.8172 -
accuracy: 0.3380 - val_loss: 2.8307 - val_accuracy: 0.1060
Epoch 2/20
141/141 [==============================] - 1s 9ms/step - loss: 1.3395 -
```

```
accuracy: 0.5089 - val_loss: 3.2411 - val_accuracy: 0.1680
Epoch 3/20
141/141 [==============================] - 1s 7ms/step - loss: 1.1227 -
accuracy: 0.5958 - val_loss: 2.2395 - val_accuracy: 0.2900
Epoch 4/20
141/141 [==============================] - 1s 7ms/step - loss: 0.9691 -
accuracy: 0.6582 - val_loss: 1.5108 - val_accuracy: 0.4720
Epoch 5/20
141/141 [==============================] - 1s 7ms/step - loss: 0.8469 -
accuracy: 0.7169 - val_loss: 1.3289 - val_accuracy: 0.5520
Epoch 6/20
141/141 [==============================] - 1s 7ms/step - loss: 0.7444 -
accuracy: 0.7638 - val_loss: 1.3557 - val_accuracy: 0.5360
Epoch 7/20
141/141 [==============================] - 1s 7ms/step - loss: 0.6531 -
accuracy: 0.8058 - val_loss: 1.3614 - val_accuracy: 0.5520
Epoch 8/20
141/141 [==============================] - 1s 7ms/step - loss: 0.5696 -
accuracy: 0.8438 - val_loss: 1.4007 - val_accuracy: 0.5420
Epoch 9/20
141/141 [==============================] - 1s 7ms/step - loss: 0.4970 -
accuracy: 0.8738 - val_loss: 1.4979 - val_accuracy: 0.5180
Epoch 10/20
141/141 [==============================] - 1s 6ms/step - loss: 0.4394 -
accuracy: 0.8938 - val_loss: 1.6191 - val_accuracy: 0.5200
Epoch 11/20
141/141 [==============================] - 1s 8ms/step - loss: 0.3891 -
accuracy: 0.9113 - val_loss: 2.0143 - val_accuracy: 0.4480
Epoch 12/20
141/141 [==============================] - 1s 10ms/step - loss: 0.3448 -
accuracy: 0.9344 - val_loss: 2.4209 - val_accuracy: 0.4020
Epoch 13/20
141/141 [==============================] - 1s 7ms/step - loss: 0.3029 -
accuracy: 0.9460 - val_loss: 2.4210 - val_accuracy: 0.4380
Epoch 14/20
141/141 [==============================] - 1s 7ms/step - loss: 0.2607 -
accuracy: 0.9649 - val_loss: 2.1427 - val_accuracy: 0.4680
Epoch 15/20
141/141 [==============================] - 1s 7ms/step - loss: 0.2190 -
accuracy: 0.9764 - val_loss: 1.7753 - val_accuracy: 0.4940
Epoch 16/20
141/141 [==============================] - 1s 7ms/step - loss: 0.1865 -
accuracy: 0.9871 - val_loss: 1.5216 - val_accuracy: 0.5520
Epoch 17/20
141/141 [==============================] - 1s 7ms/step - loss: 0.1598 -
accuracy: 0.9933 - val_loss: 1.4127 - val_accuracy: 0.5760
Epoch 18/20
141/141 [==============================] - 1s 7ms/step - loss: 0.1373 -
```

```
accuracy: 0.9964 - val_loss: 1.4383 - val_accuracy: 0.5740
Epoch 19/20
141/141 [==============================] - 1s 7ms/step - loss: 0.1188 -
accuracy: 0.9980 - val_loss: 1.4569 - val_accuracy: 0.5700
Epoch 20/20
141/141 [==============================] - 1s 7ms/step - loss: 0.1036 -
accuracy: 0.9989 - val_loss: 1.4557 - val_accuracy: 0.5740
```

[ ]:

```
my_network_no_skip = YourModel(name='network1',
                     feature_maps=32,
                     num_classes=data_manager.n_classes,
                     num_blocks=3,
                     drop_rate=0.0,
                     batch_norm=True,
                     use_skip=False,
                     optimizer='sgd',
                     learning_rate=0.001)
my_network_no_skip.build_cnn()
my_network_no_skip.summary()
```

```
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_2 (InputLayer)        [(None, 32, 32, 3)]       0

 conv2d_10 (Conv2D)          (None, 32, 32, 32)        896

 batch_normalization_6 (Batc  (None, 32, 32, 32)       128
 hNormalization)

 activation_6 (Activation)   (None, 32, 32, 32)        0

 conv2d_11 (Conv2D)          (None, 32, 32, 32)        9248

 batch_normalization_7 (Batc  (None, 32, 32, 32)       128
 hNormalization)

 activation_7 (Activation)   (None, 32, 32, 32)        0

 average_pooling2d_5 (Averag  (None, 16, 16, 32)       0
 ePooling2D)

 conv2d_12 (Conv2D)          (None, 16, 16, 64)        18496

 batch_normalization_8 (Batc  (None, 16, 16, 64)       256
 hNormalization)
```

```
 activation_8 (Activation)   (None, 16, 16, 64)        0

 conv2d_13 (Conv2D)          (None, 16, 16, 64)        36928

 batch_normalization_9 (Batc  (None, 16, 16, 64)       256
 hNormalization)

 activation_9 (Activation)   (None, 16, 16, 64)        0

 average_pooling2d_6 (Averag  (None, 8, 8, 64)         0
 ePooling2D)

 conv2d_14 (Conv2D)          (None, 8, 8, 128)         73856

 batch_normalization_10 (Bat  (None, 8, 8, 128)        512
 chNormalization)

 activation_10 (Activation)  (None, 8, 8, 128)         0

 conv2d_15 (Conv2D)          (None, 8, 8, 128)         147584

 batch_normalization_11 (Bat  (None, 8, 8, 128)        512
 chNormalization)

 activation_11 (Activation)  (None, 8, 8, 128)         0

 average_pooling2d_7 (Averag  (None, 4, 4, 128)        0
 ePooling2D)

 flatten_2 (Flatten)         (None, 2048)              0

 dense_2 (Dense)             (None, 10)                20490

=================================================================
Total params: 309,290
Trainable params: 308,394
Non-trainable params: 896
_________________________________________________________________
None
```

[ ]:
```
my_network_no_skip.fit(x_train_batch, x_val_batch, num_epochs=20)
```

```
Epoch 1/20
141/141 [==============================] - 3s 9ms/step - loss: 1.7145 -
accuracy: 0.3591 - val_loss: 2.5450 - val_accuracy: 0.1120
Epoch 2/20
141/141 [==============================] - 1s 6ms/step - loss: 1.3194 -
accuracy: 0.5147 - val_loss: 2.6301 - val_accuracy: 0.1600
```

```
Epoch 3/20
141/141 [==============================] - 1s 7ms/step - loss: 1.1367 -
accuracy: 0.5996 - val_loss: 1.9719 - val_accuracy: 0.3020
Epoch 4/20
141/141 [==============================] - 1s 6ms/step - loss: 0.9927 -
accuracy: 0.6609 - val_loss: 1.5810 - val_accuracy: 0.4420
Epoch 5/20
141/141 [==============================] - 1s 6ms/step - loss: 0.8759 -
accuracy: 0.7138 - val_loss: 1.5433 - val_accuracy: 0.4580
Epoch 6/20
141/141 [==============================] - 1s 7ms/step - loss: 0.7781 -
accuracy: 0.7636 - val_loss: 1.9402 - val_accuracy: 0.4120
Epoch 7/20
141/141 [==============================] - 1s 8ms/step - loss: 0.6990 -
accuracy: 0.7980 - val_loss: 2.0749 - val_accuracy: 0.4020
Epoch 8/20
141/141 [==============================] - 1s 9ms/step - loss: 0.6278 -
accuracy: 0.8309 - val_loss: 2.0775 - val_accuracy: 0.4140
Epoch 9/20
141/141 [==============================] - 1s 7ms/step - loss: 0.5581 -
accuracy: 0.8598 - val_loss: 2.0063 - val_accuracy: 0.4400
Epoch 10/20
141/141 [==============================] - 1s 7ms/step - loss: 0.4959 -
accuracy: 0.8862 - val_loss: 2.0783 - val_accuracy: 0.4420
Epoch 11/20
141/141 [==============================] - 1s 6ms/step - loss: 0.4391 -
accuracy: 0.9091 - val_loss: 2.1168 - val_accuracy: 0.4320
Epoch 12/20
141/141 [==============================] - 1s 7ms/step - loss: 0.3874 -
accuracy: 0.9307 - val_loss: 1.9595 - val_accuracy: 0.4360
Epoch 13/20
141/141 [==============================] - 1s 7ms/step - loss: 0.3393 -
accuracy: 0.9467 - val_loss: 1.7357 - val_accuracy: 0.4820
Epoch 14/20
141/141 [==============================] - 1s 7ms/step - loss: 0.2954 -
accuracy: 0.9613 - val_loss: 1.6131 - val_accuracy: 0.5120
Epoch 15/20
141/141 [==============================] - 1s 7ms/step - loss: 0.2531 -
accuracy: 0.9744 - val_loss: 1.6467 - val_accuracy: 0.5040
Epoch 16/20
141/141 [==============================] - 1s 7ms/step - loss: 0.2167 -
accuracy: 0.9844 - val_loss: 1.5854 - val_accuracy: 0.5160
Epoch 17/20
141/141 [==============================] - 1s 7ms/step - loss: 0.1854 -
accuracy: 0.9918 - val_loss: 1.4717 - val_accuracy: 0.5200
Epoch 18/20
141/141 [==============================] - 1s 9ms/step - loss: 0.1580 -
accuracy: 0.9951 - val_loss: 1.3932 - val_accuracy: 0.5460
```

```
Epoch 19/20
141/141 [==============================] - 1s 8ms/step - loss: 0.1354 -
accuracy: 0.9976 - val_loss: 1.3732 - val_accuracy: 0.5460
Epoch 20/20
141/141 [==============================] - 1s 7ms/step - loss: 0.1174 -
accuracy: 0.9984 - val_loss: 1.3709 - val_accuracy: 0.5680
```

The new model works apparently much much better. The originnal accuracy was only around 0.422

with skip, the accuracy becomes 0.612 impressive without skip, the accuracy becomes 0.564

It improves, but seems not much. And apparently, there is overfitting issue as accuracy is much higher than the val_accuracy

0.1.4 Question 3.4: Tune hyperparameters with grid search

Now, let us tune the $num\_blocks \in \{2, 3, 4\}$, $use\_skip \in \{True, False\}$, and $learning\_rate \in \{0.001, 0.0001\}$. Write your code for this tuning and report the result of the best model on the testing set. Note that you need to show your code for tuning and evaluating on the test set to earn the full marks. During tuning, you can set the instance variable `verbose` of your model to `False` for not showing the training details of each epoch.

[4 points]

 # REPORT THE BEST PARAMETERS AND THE TESTING ACCURACY HERE

…..

[ ]:
```
# YOU ARE REQUIRED TO INSERT YOUR CODES IN THIS CELL
# You can add more cells if necessary
best_accuracy = 0
best_model = None
best_params = {}



#set up different parameters for number_blocks, skip, and learning rate
for blocks in [2, 3, 4]:
    for skip in [True, False]:
        for lr in [0.001, 0.0001]:
            print(f"Training model with num_blocks={blocks}, use_skip={skip},␣
 ↪and learning_rate={lr}")

            model_3_4=YourModel(name='q3_4',
                     feature_maps=32,
                     num_classes=data_manager.n_classes,
                     num_blocks=blocks,
                     drop_rate=0.0,
                     batch_norm=True,
                     use_skip=skip,
                     optimizer='sgd',
```

```
                    learning_rate=lr)



            model_3_4.build_cnn()
            model_3_4.fit(x_train_batch, x_val_batch,num_epochs=20)

            val_accuracy = max(model_3_4.history.history['val_accuracy'])
            if val_accuracy > best_accuracy:
                best_accuracy = val_accuracy
                best_model = model_3_4
                best_params = {
                    "num_blocks": blocks,
                    "use_skip": skip,
                    "learning_rate": lr
                }

print(f"Best validation accuracy is {best_accuracy} with params: {best_params}")
```

```
Training model with num_blocks=2, use_skip=True, and learning_rate=0.001
Epoch 1/20
141/141 [==============================] - 3s 8ms/step - loss: 1.9165 -
accuracy: 0.3236 - val_loss: 2.7488 - val_accuracy: 0.0920
Epoch 2/20
141/141 [==============================] - 1s 6ms/step - loss: 1.3802 -
accuracy: 0.4924 - val_loss: 2.7620 - val_accuracy: 0.1980
Epoch 3/20
141/141 [==============================] - 1s 5ms/step - loss: 1.1641 -
accuracy: 0.5789 - val_loss: 2.1526 - val_accuracy: 0.3280
Epoch 4/20
141/141 [==============================] - 1s 5ms/step - loss: 1.0179 -
accuracy: 0.6398 - val_loss: 1.7824 - val_accuracy: 0.4180
Epoch 5/20
141/141 [==============================] - 1s 5ms/step - loss: 0.9008 -
accuracy: 0.6876 - val_loss: 1.6467 - val_accuracy: 0.4600
Epoch 6/20
141/141 [==============================] - 1s 5ms/step - loss: 0.8078 -
accuracy: 0.7273 - val_loss: 1.7132 - val_accuracy: 0.4660
Epoch 7/20
141/141 [==============================] - 1s 5ms/step - loss: 0.7297 -
accuracy: 0.7631 - val_loss: 1.9207 - val_accuracy: 0.4160
Epoch 8/20
141/141 [==============================] - 1s 5ms/step - loss: 0.6576 -
accuracy: 0.7907 - val_loss: 1.9553 - val_accuracy: 0.4280
Epoch 9/20
141/141 [==============================] - 1s 5ms/step - loss: 0.5889 -
accuracy: 0.8233 - val_loss: 2.1519 - val_accuracy: 0.4160
```

```
Epoch 10/20
141/141 [==============================] - 1s 6ms/step - loss: 0.5281 -
accuracy: 0.8460 - val_loss: 1.8633 - val_accuracy: 0.4640
Epoch 11/20
141/141 [==============================] - 1s 7ms/step - loss: 0.4728 -
accuracy: 0.8691 - val_loss: 1.7808 - val_accuracy: 0.4920
Epoch 12/20
141/141 [==============================] - 1s 7ms/step - loss: 0.4238 -
accuracy: 0.8920 - val_loss: 1.9491 - val_accuracy: 0.4660
Epoch 13/20
141/141 [==============================] - 1s 5ms/step - loss: 0.3834 -
accuracy: 0.9131 - val_loss: 2.0269 - val_accuracy: 0.4840
Epoch 14/20
141/141 [==============================] - 1s 5ms/step - loss: 0.3549 -
accuracy: 0.9233 - val_loss: 1.8934 - val_accuracy: 0.4880
Epoch 15/20
141/141 [==============================] - 1s 5ms/step - loss: 0.3300 -
accuracy: 0.9322 - val_loss: 1.7375 - val_accuracy: 0.5140
Epoch 16/20
141/141 [==============================] - 1s 6ms/step - loss: 0.3065 -
accuracy: 0.9407 - val_loss: 1.6873 - val_accuracy: 0.5260
Epoch 17/20
141/141 [==============================] - 1s 5ms/step - loss: 0.2790 -
accuracy: 0.9480 - val_loss: 1.7562 - val_accuracy: 0.5060
Epoch 18/20
141/141 [==============================] - 1s 5ms/step - loss: 0.2550 -
accuracy: 0.9569 - val_loss: 1.7542 - val_accuracy: 0.5100
Epoch 19/20
141/141 [==============================] - 1s 5ms/step - loss: 0.2329 -
accuracy: 0.9649 - val_loss: 1.6554 - val_accuracy: 0.5280
Epoch 20/20
141/141 [==============================] - 1s 6ms/step - loss: 0.2136 -
accuracy: 0.9729 - val_loss: 1.6399 - val_accuracy: 0.5220
Training model with num_blocks=2, use_skip=True, and learning_rate=0.0001
Epoch 1/20
141/141 [==============================] - 2s 8ms/step - loss: 2.1670 -
accuracy: 0.2229 - val_loss: 2.3032 - val_accuracy: 0.1140
Epoch 2/20
141/141 [==============================] - 1s 6ms/step - loss: 1.8024 -
accuracy: 0.3469 - val_loss: 2.3262 - val_accuracy: 0.1840
Epoch 3/20
141/141 [==============================] - 1s 6ms/step - loss: 1.6595 -
accuracy: 0.3978 - val_loss: 1.9562 - val_accuracy: 0.3120
Epoch 4/20
141/141 [==============================] - 1s 5ms/step - loss: 1.5599 -
accuracy: 0.4360 - val_loss: 1.6757 - val_accuracy: 0.3920
Epoch 5/20
141/141 [==============================] - 1s 5ms/step - loss: 1.4812 -
```

```
accuracy: 0.4640 - val_loss: 1.5819 - val_accuracy: 0.4080
Epoch 6/20
141/141 [==============================] - 1s 5ms/step - loss: 1.4160 -
accuracy: 0.4893 - val_loss: 1.5388 - val_accuracy: 0.4320
Epoch 7/20
141/141 [==============================] - 1s 5ms/step - loss: 1.3604 -
accuracy: 0.5122 - val_loss: 1.5037 - val_accuracy: 0.4620
Epoch 8/20
141/141 [==============================] - 1s 5ms/step - loss: 1.3117 -
accuracy: 0.5329 - val_loss: 1.4731 - val_accuracy: 0.4700
Epoch 9/20
141/141 [==============================] - 1s 5ms/step - loss: 1.2681 -
accuracy: 0.5471 - val_loss: 1.4462 - val_accuracy: 0.4820
Epoch 10/20
141/141 [==============================] - 1s 5ms/step - loss: 1.2287 -
accuracy: 0.5616 - val_loss: 1.4231 - val_accuracy: 0.4880
Epoch 11/20
141/141 [==============================] - 1s 6ms/step - loss: 1.1928 -
accuracy: 0.5796 - val_loss: 1.4003 - val_accuracy: 0.4920
Epoch 12/20
141/141 [==============================] - 1s 8ms/step - loss: 1.1597 -
accuracy: 0.5949 - val_loss: 1.3824 - val_accuracy: 0.5120
Epoch 13/20
141/141 [==============================] - 1s 6ms/step - loss: 1.1290 -
accuracy: 0.6102 - val_loss: 1.3665 - val_accuracy: 0.5240
Epoch 14/20
141/141 [==============================] - 1s 6ms/step - loss: 1.1004 -
accuracy: 0.6220 - val_loss: 1.3530 - val_accuracy: 0.5240
Epoch 15/20
141/141 [==============================] - 1s 5ms/step - loss: 1.0735 -
accuracy: 0.6313 - val_loss: 1.3416 - val_accuracy: 0.5260
Epoch 16/20
141/141 [==============================] - 1s 5ms/step - loss: 1.0480 -
accuracy: 0.6391 - val_loss: 1.3307 - val_accuracy: 0.5260
Epoch 17/20
141/141 [==============================] - 1s 6ms/step - loss: 1.0237 -
accuracy: 0.6489 - val_loss: 1.3217 - val_accuracy: 0.5340
Epoch 18/20
141/141 [==============================] - 1s 6ms/step - loss: 1.0007 -
accuracy: 0.6618 - val_loss: 1.3138 - val_accuracy: 0.5400
Epoch 19/20
141/141 [==============================] - 1s 5ms/step - loss: 0.9787 -
accuracy: 0.6702 - val_loss: 1.3069 - val_accuracy: 0.5400
Epoch 20/20
141/141 [==============================] - 1s 5ms/step - loss: 0.9578 -
accuracy: 0.6791 - val_loss: 1.2993 - val_accuracy: 0.5440
Training model with num_blocks=2, use_skip=False, and learning_rate=0.001
Epoch 1/20
```

```
141/141 [==============================] - 2s 7ms/step - loss: 1.7959 -
accuracy: 0.3511 - val_loss: 2.4202 - val_accuracy: 0.1260
Epoch 2/20
141/141 [==============================] - 1s 8ms/step - loss: 1.3637 -
accuracy: 0.5091 - val_loss: 2.2538 - val_accuracy: 0.2160
Epoch 3/20
141/141 [==============================] - 1s 6ms/step - loss: 1.1944 -
accuracy: 0.5742 - val_loss: 1.6436 - val_accuracy: 0.3920
Epoch 4/20
141/141 [==============================] - 1s 5ms/step - loss: 1.0677 -
accuracy: 0.6256 - val_loss: 1.4147 - val_accuracy: 0.4900
Epoch 5/20
141/141 [==============================] - 1s 6ms/step - loss: 0.9632 -
accuracy: 0.6787 - val_loss: 1.3391 - val_accuracy: 0.5260
Epoch 6/20
141/141 [==============================] - 1s 5ms/step - loss: 0.8723 -
accuracy: 0.7136 - val_loss: 1.3621 - val_accuracy: 0.5080
Epoch 7/20
141/141 [==============================] - 1s 5ms/step - loss: 0.7920 -
accuracy: 0.7524 - val_loss: 1.4365 - val_accuracy: 0.4960
Epoch 8/20
141/141 [==============================] - 1s 5ms/step - loss: 0.7195 -
accuracy: 0.7882 - val_loss: 1.4109 - val_accuracy: 0.5100
Epoch 9/20
141/141 [==============================] - 1s 5ms/step - loss: 0.6544 -
accuracy: 0.8140 - val_loss: 1.4271 - val_accuracy: 0.5020
Epoch 10/20
141/141 [==============================] - 1s 6ms/step - loss: 0.5962 -
accuracy: 0.8369 - val_loss: 1.4128 - val_accuracy: 0.5040
Epoch 11/20
141/141 [==============================] - 1s 5ms/step - loss: 0.5429 -
accuracy: 0.8607 - val_loss: 1.3887 - val_accuracy: 0.5160
Epoch 12/20
141/141 [==============================] - 1s 6ms/step - loss: 0.4942 -
accuracy: 0.8851 - val_loss: 1.3566 - val_accuracy: 0.5360
Epoch 13/20
141/141 [==============================] - 1s 7ms/step - loss: 0.4492 -
accuracy: 0.9027 - val_loss: 1.3395 - val_accuracy: 0.5480
Epoch 14/20
141/141 [==============================] - 1s 6ms/step - loss: 0.4090 -
accuracy: 0.9191 - val_loss: 1.3364 - val_accuracy: 0.5620
Epoch 15/20
141/141 [==============================] - 1s 5ms/step - loss: 0.3740 -
accuracy: 0.9309 - val_loss: 1.3676 - val_accuracy: 0.5600
Epoch 16/20
141/141 [==============================] - 1s 5ms/step - loss: 0.3403 -
accuracy: 0.9449 - val_loss: 1.3567 - val_accuracy: 0.5540
Epoch 17/20
```

```
141/141 [==============================] - 1s 5ms/step - loss: 0.3129 -
accuracy: 0.9540 - val_loss: 1.3934 - val_accuracy: 0.5400
Epoch 18/20
141/141 [==============================] - 1s 6ms/step - loss: 0.2879 -
accuracy: 0.9613 - val_loss: 1.4009 - val_accuracy: 0.5380
Epoch 19/20
141/141 [==============================] - 1s 5ms/step - loss: 0.2654 -
accuracy: 0.9709 - val_loss: 1.4073 - val_accuracy: 0.5500
Epoch 20/20
141/141 [==============================] - 1s 5ms/step - loss: 0.2425 -
accuracy: 0.9762 - val_loss: 1.3865 - val_accuracy: 0.5580
Training model with num_blocks=2, use_skip=False, and learning_rate=0.0001
Epoch 1/20
141/141 [==============================] - 2s 7ms/step - loss: 2.2027 -
accuracy: 0.2073 - val_loss: 2.3339 - val_accuracy: 0.0960
Epoch 2/20
141/141 [==============================] - 1s 7ms/step - loss: 1.8676 -
accuracy: 0.3216 - val_loss: 2.2317 - val_accuracy: 0.1620
Epoch 3/20
141/141 [==============================] - 1s 6ms/step - loss: 1.7387 -
accuracy: 0.3762 - val_loss: 1.9520 - val_accuracy: 0.2960
Epoch 4/20
141/141 [==============================] - 1s 5ms/step - loss: 1.6540 -
accuracy: 0.4116 - val_loss: 1.7425 - val_accuracy: 0.3540
Epoch 5/20
141/141 [==============================] - 1s 6ms/step - loss: 1.5878 -
accuracy: 0.4349 - val_loss: 1.6662 - val_accuracy: 0.3840
Epoch 6/20
141/141 [==============================] - 1s 6ms/step - loss: 1.5322 -
accuracy: 0.4582 - val_loss: 1.6237 - val_accuracy: 0.4100
Epoch 7/20
141/141 [==============================] - 1s 5ms/step - loss: 1.4841 -
accuracy: 0.4769 - val_loss: 1.5901 - val_accuracy: 0.4240
Epoch 8/20
141/141 [==============================] - 1s 5ms/step - loss: 1.4418 -
accuracy: 0.4936 - val_loss: 1.5582 - val_accuracy: 0.4240
Epoch 9/20
141/141 [==============================] - 1s 5ms/step - loss: 1.4034 -
accuracy: 0.5113 - val_loss: 1.5300 - val_accuracy: 0.4400
Epoch 10/20
141/141 [==============================] - 1s 5ms/step - loss: 1.3680 -
accuracy: 0.5251 - val_loss: 1.5036 - val_accuracy: 0.4500
Epoch 11/20
141/141 [==============================] - 1s 6ms/step - loss: 1.3355 -
accuracy: 0.5402 - val_loss: 1.4833 - val_accuracy: 0.4540
Epoch 12/20
141/141 [==============================] - 1s 5ms/step - loss: 1.3054 -
accuracy: 0.5549 - val_loss: 1.4634 - val_accuracy: 0.4640
```

```
Epoch 13/20
141/141 [==============================] - 1s 7ms/step - loss: 1.2772 -
accuracy: 0.5622 - val_loss: 1.4443 - val_accuracy: 0.4700
Epoch 14/20
141/141 [==============================] - 1s 7ms/step - loss: 1.2508 -
accuracy: 0.5718 - val_loss: 1.4279 - val_accuracy: 0.4780
Epoch 15/20
141/141 [==============================] - 1s 5ms/step - loss: 1.2258 -
accuracy: 0.5809 - val_loss: 1.4139 - val_accuracy: 0.4860
Epoch 16/20
141/141 [==============================] - 1s 6ms/step - loss: 1.2022 -
accuracy: 0.5924 - val_loss: 1.4026 - val_accuracy: 0.4920
Epoch 17/20
141/141 [==============================] - 1s 5ms/step - loss: 1.1799 -
accuracy: 0.6020 - val_loss: 1.3903 - val_accuracy: 0.5040
Epoch 18/20
141/141 [==============================] - 1s 5ms/step - loss: 1.1586 -
accuracy: 0.6102 - val_loss: 1.3763 - val_accuracy: 0.5040
Epoch 19/20
141/141 [==============================] - 1s 5ms/step - loss: 1.1381 -
accuracy: 0.6187 - val_loss: 1.3666 - val_accuracy: 0.5140
Epoch 20/20
141/141 [==============================] - 1s 5ms/step - loss: 1.1185 -
accuracy: 0.6267 - val_loss: 1.3551 - val_accuracy: 0.5120
Training model with num_blocks=3, use_skip=True, and learning_rate=0.001
Epoch 1/20
141/141 [==============================] - 3s 8ms/step - loss: 1.7995 -
accuracy: 0.3520 - val_loss: 2.7965 - val_accuracy: 0.1380
Epoch 2/20
141/141 [==============================] - 1s 7ms/step - loss: 1.3253 -
accuracy: 0.5176 - val_loss: 3.0005 - val_accuracy: 0.2200
Epoch 3/20
141/141 [==============================] - 1s 8ms/step - loss: 1.1307 -
accuracy: 0.5900 - val_loss: 2.1731 - val_accuracy: 0.3180
Epoch 4/20
141/141 [==============================] - 1s 9ms/step - loss: 0.9842 -
accuracy: 0.6544 - val_loss: 1.5053 - val_accuracy: 0.4600
Epoch 5/20
141/141 [==============================] - 1s 7ms/step - loss: 0.8627 -
accuracy: 0.7124 - val_loss: 1.3924 - val_accuracy: 0.4860
Epoch 6/20
141/141 [==============================] - 1s 7ms/step - loss: 0.7649 -
accuracy: 0.7527 - val_loss: 1.5152 - val_accuracy: 0.4720
Epoch 7/20
141/141 [==============================] - 1s 7ms/step - loss: 0.6734 -
accuracy: 0.7929 - val_loss: 1.6826 - val_accuracy: 0.4540
Epoch 8/20
141/141 [==============================] - 1s 7ms/step - loss: 0.5861 -
```

```
accuracy: 0.8293 - val_loss: 1.6894 - val_accuracy: 0.4420
Epoch 9/20
141/141 [==============================] - 1s 7ms/step - loss: 0.5076 -
accuracy: 0.8687 - val_loss: 1.5566 - val_accuracy: 0.4820
Epoch 10/20
141/141 [==============================] - 1s 7ms/step - loss: 0.4415 -
accuracy: 0.8998 - val_loss: 1.5980 - val_accuracy: 0.4820
Epoch 11/20
141/141 [==============================] - 1s 7ms/step - loss: 0.3833 -
accuracy: 0.9176 - val_loss: 1.5905 - val_accuracy: 0.4900
Epoch 12/20
141/141 [==============================] - 1s 7ms/step - loss: 0.3310 -
accuracy: 0.9424 - val_loss: 1.4653 - val_accuracy: 0.5420
Epoch 13/20
141/141 [==============================] - 1s 9ms/step - loss: 0.2898 -
accuracy: 0.9573 - val_loss: 1.4417 - val_accuracy: 0.5500
Epoch 14/20
141/141 [==============================] - 1s 7ms/step - loss: 0.2537 -
accuracy: 0.9678 - val_loss: 1.4241 - val_accuracy: 0.5500
Epoch 15/20
141/141 [==============================] - 1s 7ms/step - loss: 0.2234 -
accuracy: 0.9760 - val_loss: 1.5265 - val_accuracy: 0.5420
Epoch 16/20
141/141 [==============================] - 1s 7ms/step - loss: 0.1947 -
accuracy: 0.9822 - val_loss: 1.6417 - val_accuracy: 0.5260
Epoch 17/20
141/141 [==============================] - 1s 7ms/step - loss: 0.1692 -
accuracy: 0.9878 - val_loss: 1.6998 - val_accuracy: 0.5100
Epoch 18/20
141/141 [==============================] - 1s 7ms/step - loss: 0.1464 -
accuracy: 0.9936 - val_loss: 1.4843 - val_accuracy: 0.5500
Epoch 19/20
141/141 [==============================] - 1s 7ms/step - loss: 0.1236 -
accuracy: 0.9969 - val_loss: 1.3803 - val_accuracy: 0.5860
Epoch 20/20
141/141 [==============================] - 1s 7ms/step - loss: 0.1045 -
accuracy: 0.9980 - val_loss: 1.3364 - val_accuracy: 0.6160
Training model with num_blocks=3, use_skip=True, and learning_rate=0.0001
Epoch 1/20
141/141 [==============================] - 4s 13ms/step - loss: 2.2147 -
accuracy: 0.2167 - val_loss: 2.3670 - val_accuracy: 0.1000
Epoch 2/20
141/141 [==============================] - 1s 7ms/step - loss: 1.7773 -
accuracy: 0.3569 - val_loss: 2.3118 - val_accuracy: 0.1840
Epoch 3/20
141/141 [==============================] - 1s 7ms/step - loss: 1.6247 -
accuracy: 0.4164 - val_loss: 1.9523 - val_accuracy: 0.2760
Epoch 4/20
```

```
141/141 [==============================] - 1s 7ms/step - loss: 1.5251 -
accuracy: 0.4567 - val_loss: 1.6289 - val_accuracy: 0.3940
Epoch 5/20
141/141 [==============================] - 1s 7ms/step - loss: 1.4491 -
accuracy: 0.4842 - val_loss: 1.5233 - val_accuracy: 0.4380
Epoch 6/20
141/141 [==============================] - 1s 7ms/step - loss: 1.3862 -
accuracy: 0.5062 - val_loss: 1.4841 - val_accuracy: 0.4440
Epoch 7/20
141/141 [==============================] - 1s 7ms/step - loss: 1.3328 -
accuracy: 0.5289 - val_loss: 1.4552 - val_accuracy: 0.4420
Epoch 8/20
141/141 [==============================] - 1s 7ms/step - loss: 1.2861 -
accuracy: 0.5473 - val_loss: 1.4340 - val_accuracy: 0.4440
Epoch 9/20
141/141 [==============================] - 1s 7ms/step - loss: 1.2443 -
accuracy: 0.5618 - val_loss: 1.4127 - val_accuracy: 0.4740
Epoch 10/20
141/141 [==============================] - 1s 7ms/step - loss: 1.2066 -
accuracy: 0.5747 - val_loss: 1.3934 - val_accuracy: 0.4780
Epoch 11/20
141/141 [==============================] - 1s 10ms/step - loss: 1.1719 -
accuracy: 0.5896 - val_loss: 1.3751 - val_accuracy: 0.4820
Epoch 12/20
141/141 [==============================] - 1s 7ms/step - loss: 1.1399 -
accuracy: 0.6062 - val_loss: 1.3582 - val_accuracy: 0.4880
Epoch 13/20
141/141 [==============================] - 1s 7ms/step - loss: 1.1103 -
accuracy: 0.6189 - val_loss: 1.3473 - val_accuracy: 0.4900
Epoch 14/20
141/141 [==============================] - 1s 7ms/step - loss: 1.0821 -
accuracy: 0.6353 - val_loss: 1.3325 - val_accuracy: 0.4960
Epoch 15/20
141/141 [==============================] - 1s 7ms/step - loss: 1.0557 -
accuracy: 0.6473 - val_loss: 1.3231 - val_accuracy: 0.5040
Epoch 16/20
141/141 [==============================] - 1s 7ms/step - loss: 1.0307 -
accuracy: 0.6578 - val_loss: 1.3144 - val_accuracy: 0.5000
Epoch 17/20
141/141 [==============================] - 1s 7ms/step - loss: 1.0068 -
accuracy: 0.6702 - val_loss: 1.3044 - val_accuracy: 0.5140
Epoch 18/20
141/141 [==============================] - 1s 7ms/step - loss: 0.9839 -
accuracy: 0.6796 - val_loss: 1.2964 - val_accuracy: 0.5140
Epoch 19/20
141/141 [==============================] - 1s 7ms/step - loss: 0.9617 -
accuracy: 0.6900 - val_loss: 1.2892 - val_accuracy: 0.5220
Epoch 20/20
```

```
141/141 [==============================] - 1s 9ms/step - loss: 0.9404 -
accuracy: 0.7013 - val_loss: 1.2806 - val_accuracy: 0.5240
Training model with num_blocks=3, use_skip=False, and learning_rate=0.001
Epoch 1/20
141/141 [==============================] - 3s 9ms/step - loss: 1.7919 -
accuracy: 0.3500 - val_loss: 2.6187 - val_accuracy: 0.1320
Epoch 2/20
141/141 [==============================] - 1s 7ms/step - loss: 1.3576 -
accuracy: 0.5151 - val_loss: 2.5274 - val_accuracy: 0.1800
Epoch 3/20
141/141 [==============================] - 1s 7ms/step - loss: 1.1632 -
accuracy: 0.5942 - val_loss: 1.7175 - val_accuracy: 0.3780
Epoch 4/20
141/141 [==============================] - 1s 7ms/step - loss: 1.0162 -
accuracy: 0.6573 - val_loss: 1.4810 - val_accuracy: 0.4860
Epoch 5/20
141/141 [==============================] - 1s 7ms/step - loss: 0.8937 -
accuracy: 0.7164 - val_loss: 1.6572 - val_accuracy: 0.4440
Epoch 6/20
141/141 [==============================] - 1s 9ms/step - loss: 0.7865 -
accuracy: 0.7638 - val_loss: 1.6164 - val_accuracy: 0.4520
Epoch 7/20
141/141 [==============================] - 1s 9ms/step - loss: 0.6920 -
accuracy: 0.8033 - val_loss: 1.5467 - val_accuracy: 0.4760
Epoch 8/20
141/141 [==============================] - 1s 9ms/step - loss: 0.6186 -
accuracy: 0.8382 - val_loss: 1.6237 - val_accuracy: 0.4760
Epoch 9/20
141/141 [==============================] - 1s 7ms/step - loss: 0.5561 -
accuracy: 0.8609 - val_loss: 1.4653 - val_accuracy: 0.4760
Epoch 10/20
141/141 [==============================] - 1s 7ms/step - loss: 0.5024 -
accuracy: 0.8802 - val_loss: 1.5936 - val_accuracy: 0.4720
Epoch 11/20
141/141 [==============================] - 1s 7ms/step - loss: 0.4391 -
accuracy: 0.9107 - val_loss: 1.5274 - val_accuracy: 0.4860
Epoch 12/20
141/141 [==============================] - 1s 7ms/step - loss: 0.3817 -
accuracy: 0.9327 - val_loss: 1.3972 - val_accuracy: 0.5040
Epoch 13/20
141/141 [==============================] - 1s 7ms/step - loss: 0.3332 -
accuracy: 0.9489 - val_loss: 1.3769 - val_accuracy: 0.5280
Epoch 14/20
141/141 [==============================] - 1s 7ms/step - loss: 0.2918 -
accuracy: 0.9633 - val_loss: 1.4067 - val_accuracy: 0.5020
Epoch 15/20
141/141 [==============================] - 1s 7ms/step - loss: 0.2529 -
accuracy: 0.9769 - val_loss: 1.5056 - val_accuracy: 0.4800
```

```
Epoch 16/20
141/141 [==============================] - 1s 7ms/step - loss: 0.2209 -
accuracy: 0.9842 - val_loss: 1.6467 - val_accuracy: 0.4760
Epoch 17/20
141/141 [==============================] - 1s 8ms/step - loss: 0.1952 -
accuracy: 0.9900 - val_loss: 1.6863 - val_accuracy: 0.4640
Epoch 18/20
141/141 [==============================] - 1s 9ms/step - loss: 0.1732 -
accuracy: 0.9927 - val_loss: 1.5936 - val_accuracy: 0.4860
Epoch 19/20
141/141 [==============================] - 1s 8ms/step - loss: 0.1512 -
accuracy: 0.9964 - val_loss: 1.5589 - val_accuracy: 0.4860
Epoch 20/20
141/141 [==============================] - 1s 7ms/step - loss: 0.1317 -
accuracy: 0.9982 - val_loss: 1.5157 - val_accuracy: 0.5160
Training model with num_blocks=3, use_skip=False, and learning_rate=0.0001
Epoch 1/20
141/141 [==============================] - 3s 9ms/step - loss: 2.1465 -
accuracy: 0.2236 - val_loss: 2.3016 - val_accuracy: 0.1040
Epoch 2/20
141/141 [==============================] - 1s 7ms/step - loss: 1.8046 -
accuracy: 0.3469 - val_loss: 2.2845 - val_accuracy: 0.1660
Epoch 3/20
141/141 [==============================] - 1s 7ms/step - loss: 1.6620 -
accuracy: 0.3976 - val_loss: 1.9542 - val_accuracy: 0.2960
Epoch 4/20
141/141 [==============================] - 1s 7ms/step - loss: 1.5682 -
accuracy: 0.4367 - val_loss: 1.6419 - val_accuracy: 0.4020
Epoch 5/20
141/141 [==============================] - 1s 9ms/step - loss: 1.4970 -
accuracy: 0.4664 - val_loss: 1.5449 - val_accuracy: 0.4400
Epoch 6/20
141/141 [==============================] - 1s 10ms/step - loss: 1.4399 -
accuracy: 0.4949 - val_loss: 1.4979 - val_accuracy: 0.4600
Epoch 7/20
141/141 [==============================] - 1s 7ms/step - loss: 1.3920 -
accuracy: 0.5160 - val_loss: 1.4633 - val_accuracy: 0.4740
Epoch 8/20
141/141 [==============================] - 1s 7ms/step - loss: 1.3503 -
accuracy: 0.5358 - val_loss: 1.4373 - val_accuracy: 0.4800
Epoch 9/20
141/141 [==============================] - 1s 7ms/step - loss: 1.3128 -
accuracy: 0.5533 - val_loss: 1.4142 - val_accuracy: 0.4880
Epoch 10/20
141/141 [==============================] - 1s 7ms/step - loss: 1.2784 -
accuracy: 0.5702 - val_loss: 1.3931 - val_accuracy: 0.5000
Epoch 11/20
141/141 [==============================] - 1s 7ms/step - loss: 1.2467 -
```

```
accuracy: 0.5796 - val_loss: 1.3750 - val_accuracy: 0.5020
Epoch 12/20
141/141 [==============================] - 1s 7ms/step - loss: 1.2171 -
accuracy: 0.5944 - val_loss: 1.3591 - val_accuracy: 0.5040
Epoch 13/20
141/141 [==============================] - 1s 7ms/step - loss: 1.1890 -
accuracy: 0.6016 - val_loss: 1.3464 - val_accuracy: 0.5120
Epoch 14/20
141/141 [==============================] - 1s 8ms/step - loss: 1.1625 -
accuracy: 0.6118 - val_loss: 1.3360 - val_accuracy: 0.5160
Epoch 15/20
141/141 [==============================] - 1s 9ms/step - loss: 1.1370 -
accuracy: 0.6256 - val_loss: 1.3234 - val_accuracy: 0.5200
Epoch 16/20
141/141 [==============================] - 1s 7ms/step - loss: 1.1127 -
accuracy: 0.6371 - val_loss: 1.3119 - val_accuracy: 0.5200
Epoch 17/20
141/141 [==============================] - 1s 7ms/step - loss: 1.0893 -
accuracy: 0.6502 - val_loss: 1.3022 - val_accuracy: 0.5200
Epoch 18/20
141/141 [==============================] - 1s 7ms/step - loss: 1.0668 -
accuracy: 0.6624 - val_loss: 1.2934 - val_accuracy: 0.5260
Epoch 19/20
141/141 [==============================] - 1s 7ms/step - loss: 1.0448 -
accuracy: 0.6740 - val_loss: 1.2837 - val_accuracy: 0.5280
Epoch 20/20
141/141 [==============================] - 1s 7ms/step - loss: 1.0234 -
accuracy: 0.6851 - val_loss: 1.2750 - val_accuracy: 0.5320
Training model with num_blocks=4, use_skip=True, and learning_rate=0.001
Epoch 1/20
141/141 [==============================] - 5s 15ms/step - loss: 1.8263 -
accuracy: 0.3364 - val_loss: 2.8242 - val_accuracy: 0.1120
Epoch 2/20
141/141 [==============================] - 1s 9ms/step - loss: 1.3530 -
accuracy: 0.5000 - val_loss: 3.0550 - val_accuracy: 0.1300
Epoch 3/20
141/141 [==============================] - 1s 8ms/step - loss: 1.1058 -
accuracy: 0.6002 - val_loss: 2.0405 - val_accuracy: 0.3000
Epoch 4/20
141/141 [==============================] - 1s 9ms/step - loss: 0.9042 -
accuracy: 0.6938 - val_loss: 1.6576 - val_accuracy: 0.4500
Epoch 5/20
141/141 [==============================] - 1s 9ms/step - loss: 0.7417 -
accuracy: 0.7649 - val_loss: 1.5670 - val_accuracy: 0.4640
Epoch 6/20
141/141 [==============================] - 1s 8ms/step - loss: 0.6125 -
accuracy: 0.8229 - val_loss: 1.5563 - val_accuracy: 0.4700
Epoch 7/20
```

```
141/141 [==============================] - 1s 8ms/step - loss: 0.5179 -
accuracy: 0.8671 - val_loss: 2.4837 - val_accuracy: 0.3900
Epoch 8/20
141/141 [==============================] - 1s 9ms/step - loss: 0.4439 -
accuracy: 0.8922 - val_loss: 3.8181 - val_accuracy: 0.3420
Epoch 9/20
141/141 [==============================] - 1s 8ms/step - loss: 0.3611 -
accuracy: 0.9247 - val_loss: 3.1617 - val_accuracy: 0.3700
Epoch 10/20
141/141 [==============================] - 1s 8ms/step - loss: 0.2806 -
accuracy: 0.9524 - val_loss: 2.5320 - val_accuracy: 0.4100
Epoch 11/20
141/141 [==============================] - 1s 8ms/step - loss: 0.2236 -
accuracy: 0.9689 - val_loss: 1.9231 - val_accuracy: 0.4640
Epoch 12/20
141/141 [==============================] - 1s 8ms/step - loss: 0.1874 -
accuracy: 0.9756 - val_loss: 1.7596 - val_accuracy: 0.5040
Epoch 13/20
141/141 [==============================] - 1s 8ms/step - loss: 0.1607 -
accuracy: 0.9793 - val_loss: 1.7542 - val_accuracy: 0.4820
Epoch 14/20
141/141 [==============================] - 1s 8ms/step - loss: 0.1397 -
accuracy: 0.9847 - val_loss: 2.0238 - val_accuracy: 0.4840
Epoch 15/20
141/141 [==============================] - 1s 8ms/step - loss: 0.1143 -
accuracy: 0.9880 - val_loss: 1.7279 - val_accuracy: 0.5160
Epoch 16/20
141/141 [==============================] - 1s 8ms/step - loss: 0.0941 -
accuracy: 0.9909 - val_loss: 1.7014 - val_accuracy: 0.5300
Epoch 17/20
141/141 [==============================] - 1s 10ms/step - loss: 0.0737 -
accuracy: 0.9949 - val_loss: 1.5870 - val_accuracy: 0.5700
Epoch 18/20
141/141 [==============================] - 1s 9ms/step - loss: 0.0539 -
accuracy: 0.9971 - val_loss: 1.4798 - val_accuracy: 0.5960
Epoch 19/20
141/141 [==============================] - 1s 8ms/step - loss: 0.0398 -
accuracy: 0.9991 - val_loss: 1.5578 - val_accuracy: 0.5820
Epoch 20/20
141/141 [==============================] - 1s 8ms/step - loss: 0.0286 -
accuracy: 0.9993 - val_loss: 1.6076 - val_accuracy: 0.5780
Training model with num_blocks=4, use_skip=True, and learning_rate=0.0001
Epoch 1/20
141/141 [==============================] - 4s 10ms/step - loss: 2.0299 -
accuracy: 0.2631 - val_loss: 2.4121 - val_accuracy: 0.1100
Epoch 2/20
141/141 [==============================] - 1s 8ms/step - loss: 1.6729 -
accuracy: 0.3880 - val_loss: 2.5415 - val_accuracy: 0.1120
```

```
Epoch 3/20
141/141 [==============================] - 1s 10ms/step - loss: 1.5266 -
accuracy: 0.4462 - val_loss: 1.9680 - val_accuracy: 0.2860
Epoch 4/20
141/141 [==============================] - 1s 10ms/step - loss: 1.4239 -
accuracy: 0.4851 - val_loss: 1.5731 - val_accuracy: 0.4340
Epoch 5/20
141/141 [==============================] - 1s 8ms/step - loss: 1.3428 -
accuracy: 0.5171 - val_loss: 1.4764 - val_accuracy: 0.4540
Epoch 6/20
141/141 [==============================] - 1s 8ms/step - loss: 1.2749 -
accuracy: 0.5538 - val_loss: 1.4382 - val_accuracy: 0.4760
Epoch 7/20
141/141 [==============================] - 1s 8ms/step - loss: 1.2159 -
accuracy: 0.5800 - val_loss: 1.4089 - val_accuracy: 0.4940
Epoch 8/20
141/141 [==============================] - 1s 9ms/step - loss: 1.1629 -
accuracy: 0.5991 - val_loss: 1.3883 - val_accuracy: 0.5000
Epoch 9/20
141/141 [==============================] - 1s 8ms/step - loss: 1.1132 -
accuracy: 0.6204 - val_loss: 1.3682 - val_accuracy: 0.5100
Epoch 10/20
141/141 [==============================] - 1s 8ms/step - loss: 1.0674 -
accuracy: 0.6373 - val_loss: 1.3468 - val_accuracy: 0.5220
Epoch 11/20
141/141 [==============================] - 1s 8ms/step - loss: 1.0244 -
accuracy: 0.6587 - val_loss: 1.3303 - val_accuracy: 0.5220
Epoch 12/20
141/141 [==============================] - 2s 11ms/step - loss: 0.9834 -
accuracy: 0.6767 - val_loss: 1.3169 - val_accuracy: 0.5160
Epoch 13/20
141/141 [==============================] - 1s 8ms/step - loss: 0.9445 -
accuracy: 0.6931 - val_loss: 1.2995 - val_accuracy: 0.5240
Epoch 14/20
141/141 [==============================] - 1s 8ms/step - loss: 0.9074 -
accuracy: 0.7107 - val_loss: 1.2881 - val_accuracy: 0.5320
Epoch 15/20
141/141 [==============================] - 1s 8ms/step - loss: 0.8717 -
accuracy: 0.7269 - val_loss: 1.2732 - val_accuracy: 0.5420
Epoch 16/20
141/141 [==============================] - 1s 8ms/step - loss: 0.8374 -
accuracy: 0.7478 - val_loss: 1.2640 - val_accuracy: 0.5420
Epoch 17/20
141/141 [==============================] - 1s 8ms/step - loss: 0.8041 -
accuracy: 0.7622 - val_loss: 1.2550 - val_accuracy: 0.5380
Epoch 18/20
141/141 [==============================] - 1s 8ms/step - loss: 0.7721 -
accuracy: 0.7778 - val_loss: 1.2504 - val_accuracy: 0.5480
```

```
Epoch 19/20
141/141 [==============================] - 1s 8ms/step - loss: 0.7413 -
accuracy: 0.7953 - val_loss: 1.2438 - val_accuracy: 0.5460
Epoch 20/20
141/141 [==============================] - 1s 9ms/step - loss: 0.7113 -
accuracy: 0.8122 - val_loss: 1.2378 - val_accuracy: 0.5520
Training model with num_blocks=4, use_skip=False, and learning_rate=0.001
Epoch 1/20
141/141 [==============================] - 4s 10ms/step - loss: 1.7467 -
accuracy: 0.3518 - val_loss: 2.4245 - val_accuracy: 0.1060
Epoch 2/20
141/141 [==============================] - 1s 9ms/step - loss: 1.3043 -
accuracy: 0.5218 - val_loss: 2.8335 - val_accuracy: 0.1520
Epoch 3/20
141/141 [==============================] - 1s 8ms/step - loss: 1.0767 -
accuracy: 0.6238 - val_loss: 2.1762 - val_accuracy: 0.2960
Epoch 4/20
141/141 [==============================] - 1s 8ms/step - loss: 0.9019 -
accuracy: 0.6996 - val_loss: 1.5227 - val_accuracy: 0.4480
Epoch 5/20
141/141 [==============================] - 1s 8ms/step - loss: 0.7540 -
accuracy: 0.7680 - val_loss: 1.4505 - val_accuracy: 0.4560
Epoch 6/20
141/141 [==============================] - 1s 8ms/step - loss: 0.6320 -
accuracy: 0.8216 - val_loss: 1.4151 - val_accuracy: 0.4960
Epoch 7/20
141/141 [==============================] - 1s 9ms/step - loss: 0.5225 -
accuracy: 0.8653 - val_loss: 1.5505 - val_accuracy: 0.4680
Epoch 8/20
141/141 [==============================] - 1s 10ms/step - loss: 0.4262 -
accuracy: 0.9033 - val_loss: 1.5769 - val_accuracy: 0.4700
Epoch 9/20
141/141 [==============================] - 1s 10ms/step - loss: 0.3423 -
accuracy: 0.9351 - val_loss: 1.5709 - val_accuracy: 0.4920
Epoch 10/20
141/141 [==============================] - 1s 8ms/step - loss: 0.2663 -
accuracy: 0.9640 - val_loss: 2.3646 - val_accuracy: 0.3600
Epoch 11/20
141/141 [==============================] - 1s 8ms/step - loss: 0.2170 -
accuracy: 0.9756 - val_loss: 2.5921 - val_accuracy: 0.3280
Epoch 12/20
141/141 [==============================] - 1s 8ms/step - loss: 0.1750 -
accuracy: 0.9833 - val_loss: 2.0552 - val_accuracy: 0.4300
Epoch 13/20
141/141 [==============================] - 1s 8ms/step - loss: 0.1356 -
accuracy: 0.9907 - val_loss: 1.7173 - val_accuracy: 0.5140
Epoch 14/20
141/141 [==============================] - 1s 8ms/step - loss: 0.0994 -
```

```
accuracy: 0.9953 - val_loss: 1.6295 - val_accuracy: 0.5240
Epoch 15/20
141/141 [==============================] - 1s 8ms/step - loss: 0.0753 -
accuracy: 0.9980 - val_loss: 1.7714 - val_accuracy: 0.5240
Epoch 16/20
141/141 [==============================] - 1s 8ms/step - loss: 0.0542 -
accuracy: 0.9996 - val_loss: 1.7023 - val_accuracy: 0.5520
Epoch 17/20
141/141 [==============================] - 1s 8ms/step - loss: 0.0400 -
accuracy: 0.9996 - val_loss: 1.8312 - val_accuracy: 0.5340
Epoch 18/20
141/141 [==============================] - 2s 11ms/step - loss: 0.0289 -
accuracy: 1.0000 - val_loss: 1.6550 - val_accuracy: 0.5600
Epoch 19/20
141/141 [==============================] - 1s 8ms/step - loss: 0.0212 -
accuracy: 1.0000 - val_loss: 1.5509 - val_accuracy: 0.5660
Epoch 20/20
141/141 [==============================] - 1s 8ms/step - loss: 0.0167 -
accuracy: 1.0000 - val_loss: 1.5023 - val_accuracy: 0.5720
Training model with num_blocks=4, use_skip=False, and learning_rate=0.0001
Epoch 1/20
141/141 [==============================] - 4s 11ms/step - loss: 2.1205 -
accuracy: 0.2304 - val_loss: 2.3534 - val_accuracy: 0.0880
Epoch 2/20
141/141 [==============================] - 1s 8ms/step - loss: 1.7792 -
accuracy: 0.3682 - val_loss: 2.4094 - val_accuracy: 0.1020
Epoch 3/20
141/141 [==============================] - 1s 8ms/step - loss: 1.6251 -
accuracy: 0.4247 - val_loss: 2.0627 - val_accuracy: 0.2360
Epoch 4/20
141/141 [==============================] - 1s 10ms/step - loss: 1.5212 -
accuracy: 0.4638 - val_loss: 1.7005 - val_accuracy: 0.3820
Epoch 5/20
141/141 [==============================] - 2s 12ms/step - loss: 1.4425 -
accuracy: 0.4956 - val_loss: 1.5809 - val_accuracy: 0.4340
Epoch 6/20
141/141 [==============================] - 1s 8ms/step - loss: 1.3778 -
accuracy: 0.5220 - val_loss: 1.5305 - val_accuracy: 0.4460
Epoch 7/20
141/141 [==============================] - 1s 8ms/step - loss: 1.3219 -
accuracy: 0.5456 - val_loss: 1.4987 - val_accuracy: 0.4600
Epoch 8/20
141/141 [==============================] - 1s 8ms/step - loss: 1.2712 -
accuracy: 0.5691 - val_loss: 1.4723 - val_accuracy: 0.4620
Epoch 9/20
141/141 [==============================] - 1s 9ms/step - loss: 1.2244 -
accuracy: 0.5940 - val_loss: 1.4470 - val_accuracy: 0.4720
Epoch 10/20
```

```
141/141 [==============================] - 1s 8ms/step - loss: 1.1808 -
accuracy: 0.6116 - val_loss: 1.4280 - val_accuracy: 0.4760
Epoch 11/20
141/141 [==============================] - 1s 8ms/step - loss: 1.1394 -
accuracy: 0.6302 - val_loss: 1.4109 - val_accuracy: 0.4800
Epoch 12/20
141/141 [==============================] - 1s 9ms/step - loss: 1.1003 -
accuracy: 0.6480 - val_loss: 1.3933 - val_accuracy: 0.4860
Epoch 13/20
141/141 [==============================] - 2s 11ms/step - loss: 1.0629 -
accuracy: 0.6633 - val_loss: 1.3827 - val_accuracy: 0.4920
Epoch 14/20
141/141 [==============================] - 2s 11ms/step - loss: 1.0269 -
accuracy: 0.6776 - val_loss: 1.3685 - val_accuracy: 0.4900
Epoch 15/20
141/141 [==============================] - 1s 8ms/step - loss: 0.9922 -
accuracy: 0.6960 - val_loss: 1.3606 - val_accuracy: 0.4900
Epoch 16/20
141/141 [==============================] - 1s 9ms/step - loss: 0.9585 -
accuracy: 0.7129 - val_loss: 1.3465 - val_accuracy: 0.4940
Epoch 17/20
141/141 [==============================] - 1s 9ms/step - loss: 0.9259 -
accuracy: 0.7287 - val_loss: 1.3301 - val_accuracy: 0.5000
Epoch 18/20
141/141 [==============================] - 1s 9ms/step - loss: 0.8944 -
accuracy: 0.7433 - val_loss: 1.3156 - val_accuracy: 0.5040
Epoch 19/20
141/141 [==============================] - 1s 9ms/step - loss: 0.8640 -
accuracy: 0.7558 - val_loss: 1.3010 - val_accuracy: 0.5220
Epoch 20/20
141/141 [==============================] - 1s 9ms/step - loss: 0.8345 -
accuracy: 0.7687 - val_loss: 1.2893 - val_accuracy: 0.5300
Best validation accuracy is 0.6159999966621399 with params: {'num_blocks': 3,
'use_skip': True, 'learning_rate': 0.001}
```

Result: The best validation accuracy is 0.598 with num_blocks=4, use_skip=true,learning_rate=0.001

comments: the next time I run the same code, the result changes to: num_block=3,use_skip=True,Learning_rate=0.001. validation accuracy becomes 0.616

It is noticed that the overfitting problem has been solved a bit

0.1.5 Question 3.5: Apply data augmentation

We now try to apply data augmentation to improve the performance. Extend the code of the class `YourModel` so that if the attribute `is_augmentation` is set to `True`, we apply the data augmentation. Also you need to incorporate early stopping to your training process. Specifically, you early stop the training if the valid accuracy cannot increase in three consecutive epochs.

[4 points]

```
[ ]: from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

Wtire your code in the cell below. Hint that you can rewrite the code of the `fit` method to apply the data augmentation. In addition, you can copy the code of `build_cnn` method above to reuse here.

```
[ ]: import numpy as np
class YourModel(BaseImageClassifier):
    def __init__(self,
                 name='network1',
                 width=32, height=32, depth=3,
                 num_blocks=2,
                 feature_maps=32,
                 num_classes=4,
                 drop_rate=0.2,
                 batch_norm = None,
                 is_augmentation = False,
                 activation_func='relu',
                 use_skip = True,
                 optimizer='adam',
                 batch_size=10,
                 num_epochs= 20,
                 learning_rate=0.0001):
        super(YourModel, self).__init__(name, width, height, depth, num_blocks,␣
↪feature_maps, num_classes, drop_rate, batch_norm, is_augmentation,
                                        activation_func, use_skip, optimizer,␣
↪batch_size, num_epochs, learning_rate)

    def custom_block(self, x, filters):
        # skip connection
        shortcut = x

        # First Conv layer
        x = layers.Conv2D(filters=filters, kernel_size=(3, 3), strides=(1, 1),␣
↪padding="same")(x)
        if self.batch_norm:
            x = layers.BatchNormalization()(x)
        x = layers.Activation(self.activation_func)(x)

        # Second Conv layer
        x = layers.Conv2D(filters=filters, kernel_size=(3, 3), strides=(1, 1),␣
↪padding="same")(x)
        if self.batch_norm:
            x = layers.BatchNormalization()(x)

        # Add the shortcut to the output
```

```
            if self.use_skip:
                x = tf.keras.layers.add([shortcut, x])
            x = layers.Activation(self.activation_func)(x)

            # Mean Pooling layer
            x = layers.AveragePooling2D(pool_size=(2, 2), strides=(2, 2),
↪padding="same")(x)

            # Dropout
            x = layers.Dropout(rate=self.drop_rate)(x)

            return x

    def build_cnn(self):
        input_tensor = layers.Input(shape=(self.height, self.width, self.depth))
        x = input_tensor

        for current_feature_maps in self.feature_maps:
            # Initial convolution
            x = layers.Conv2D(current_feature_maps, (3, 3), padding='same',
↪activation=None)(x)
            if self.batch_norm:
                x = layers.BatchNormalization()(x)
            x = layers.Activation(self.activation_func)(x)

            # Second convolution with skip connection
            shortcut = x
            x = layers.Conv2D(current_feature_maps, (3, 3), padding='same',
↪activation=None)(x)
            if self.batch_norm:
                x = layers.BatchNormalization()(x)
            if self.use_skip:
                x = layers.Add()([shortcut, x])
            x = layers.Activation(self.activation_func)(x)

            # Mean Pooling and Dropout
            x = layers.AveragePooling2D(pool_size=(2, 2), padding='same')(x)
            if self.drop_rate > 0:
                x = layers.Dropout(self.drop_rate)(x)

        x = layers.Flatten()(x)
        x = layers.Dense(self.num_classes, activation='softmax')(x)

        self.model = models.Model(inputs=input_tensor, outputs=x)
        self.model.compile(optimizer=self.optimizer,
↪loss='sparse_categorical_crossentropy', metrics=['accuracy'])
```

```
    def fit(self, data_manager, batch_size=None, num_epochs=None):
      batch_size = self.batch_size if batch_size is None else batch_size
      num_epochs = self.num_epochs if num_epochs is None else num_epochs

      x_train_batch = self.optimize_data_pipeline(data_manager.ds_train,␣
↪batch_size=batch_size)
      x_val_batch = self.optimize_data_pipeline(data_manager.ds_val,␣
↪batch_size=batch_size)

      if self.is_augmentation:#added code for data augmentation
        train_data_list, train_labels_list = [], []
        for data, labels in x_train_batch:
            train_data_list.append(data.numpy())
            train_labels_list.append(labels.numpy())

        train_data = np.concatenate(train_data_list, axis=0)
        train_labels = np.concatenate(train_labels_list, axis=0)

        datagen = ImageDataGenerator(rotation_range=10,
                                     width_shift_range=0.1,
                                     height_shift_range=0.1,
                                     horizontal_flip=True)#data augmentation

        datagen.fit(train_data)

        self.model.fit(datagen.flow(train_data, train_labels, batch_size=self.
↪batch_size),
                       validation_data=x_val_batch,
                       epochs=self.num_epochs,
                       verbose=self.verbose)
      else:
        self.model.fit(x_train_batch, validation_data=x_val_batch, epochs=self.
↪num_epochs, batch_size=self.batch_size, verbose=self.verbose)
```

[ ]:
```
model_3_5=YourModel(name='q3_5',
                    feature_maps=32,
                    num_classes=data_manager.n_classes,
                    num_blocks=4,
                    drop_rate=0.0,
                    batch_norm=True,
                    is_augmentation=True,
                    use_skip=True,
                    optimizer='sgd',
                    learning_rate=0.001)
```

```
model_3_5.build_cnn()
model_3_5.fit(data_manager)
```

```
Epoch 1/20
450/450 [==============================] - 9s 12ms/step - loss: 2.0184 -
accuracy: 0.3062 - val_loss: 1.7373 - val_accuracy: 0.3420
Epoch 2/20
450/450 [==============================] - 6s 14ms/step - loss: 1.6659 -
accuracy: 0.4024 - val_loss: 1.5998 - val_accuracy: 0.3980
Epoch 3/20
450/450 [==============================] - 5s 11ms/step - loss: 1.5359 -
accuracy: 0.4438 - val_loss: 1.6073 - val_accuracy: 0.3860
Epoch 4/20
450/450 [==============================] - 6s 12ms/step - loss: 1.4367 -
accuracy: 0.4691 - val_loss: 1.4128 - val_accuracy: 0.4860
Epoch 5/20
450/450 [==============================] - 6s 12ms/step - loss: 1.3578 -
accuracy: 0.5047 - val_loss: 1.3745 - val_accuracy: 0.4980
Epoch 6/20
450/450 [==============================] - 5s 12ms/step - loss: 1.2977 -
accuracy: 0.5216 - val_loss: 1.7192 - val_accuracy: 0.4480
Epoch 7/20
450/450 [==============================] - 6s 14ms/step - loss: 1.2417 -
accuracy: 0.5478 - val_loss: 1.1789 - val_accuracy: 0.5520
Epoch 8/20
450/450 [==============================] - 5s 11ms/step - loss: 1.1722 -
accuracy: 0.5644 - val_loss: 1.5165 - val_accuracy: 0.5140
Epoch 9/20
450/450 [==============================] - 5s 11ms/step - loss: 1.1578 -
accuracy: 0.5718 - val_loss: 1.5003 - val_accuracy: 0.5200
Epoch 10/20
450/450 [==============================] - 6s 13ms/step - loss: 1.0827 -
accuracy: 0.6007 - val_loss: 1.5834 - val_accuracy: 0.4760
Epoch 11/20
450/450 [==============================] - 5s 12ms/step - loss: 1.0400 -
accuracy: 0.6156 - val_loss: 1.1756 - val_accuracy: 0.5620
Epoch 12/20
450/450 [==============================] - 6s 14ms/step - loss: 1.0200 -
accuracy: 0.6300 - val_loss: 1.1522 - val_accuracy: 0.6080
Epoch 13/20
450/450 [==============================] - 5s 11ms/step - loss: 0.9788 -
accuracy: 0.6467 - val_loss: 1.1999 - val_accuracy: 0.5800
Epoch 14/20
450/450 [==============================] - 5s 12ms/step - loss: 0.9303 -
accuracy: 0.6613 - val_loss: 1.2312 - val_accuracy: 0.5840
Epoch 15/20
```

```
450/450 [==============================] - 5s 11ms/step - loss: 0.9233 -
accuracy: 0.6636 - val_loss: 1.8355 - val_accuracy: 0.4180
Epoch 16/20
450/450 [==============================] - 6s 13ms/step - loss: 0.8817 -
accuracy: 0.6767 - val_loss: 1.5634 - val_accuracy: 0.5160
Epoch 17/20
450/450 [==============================] - 5s 11ms/step - loss: 0.8544 -
accuracy: 0.6818 - val_loss: 1.3581 - val_accuracy: 0.5500
Epoch 18/20
450/450 [==============================] - 5s 12ms/step - loss: 0.8375 -
accuracy: 0.6947 - val_loss: 2.0195 - val_accuracy: 0.4120
Epoch 19/20
450/450 [==============================] - 6s 12ms/step - loss: 0.8080 -
accuracy: 0.7067 - val_loss: 1.2482 - val_accuracy: 0.5800
Epoch 20/20
450/450 [==============================] - 5s 11ms/step - loss: 0.7768 -
accuracy: 0.7142 - val_loss: 1.2149 - val_accuracy: 0.5880
```

0.1.6 Question 3.6: Observe model performance with data augmentation

Leverage your best model with the data augmentation and try to observe the difference in performance between using data augmentation and not using it.

[4 points]

From the observation, in terms of the val_accuracy, it is 0.588, not improved a lot. but I noticed that, without data augmentation, the accuracy is almost 1 while val_accuracy is only around 0.6, which meaning that there was a huge overfitting issue.

With the data augmentation, although validation accuracy does not improve a lot, the accuracy downs to 0.7142, meaning that the overfitting issue has been improved a lot

0.1.7 Question 3.7: Explore data mixup technique

[4 points]

Data mixup is another super-simple technique used to boost the generalization ability of deep learning models. You need to incoroporate data mixup technique to the above deep learning model and experiment its performance. There are some papers and documents for data mixup as follows: - Main paper for data mixup link for main paper and a good article article link.

You need to extend your model developed above, train a model using data mixup, and write your observations and comments about the result.

[ ]:
```python
# YOU ARE REQUIRED TO INSERT YOUR CODES IN THIS CELL
# You can add more cells if necessary
import numpy as np
class YourModel(BaseImageClassifier):
    def __init__(self,
                 name='network1',
                 width=32, height=32, depth=3,
```

```python
                 num_blocks=2,
                 feature_maps=32,
                 num_classes=4,
                 drop_rate=0.2,
                 batch_norm = None,
                 is_augmentation = False,
                 activation_func='relu',
                 use_skip = True,
                 optimizer='adam',
                 batch_size=10,
                 num_epochs= 20,
                 learning_rate=0.0001):
        super(YourModel, self).__init__(name, width, height, depth, num_blocks,␣
↪feature_maps, num_classes, drop_rate, batch_norm, is_augmentation,
                                       activation_func, use_skip, optimizer,␣
↪batch_size, num_epochs, learning_rate)

    def custom_block(self, x, filters):
        # skip connection
        shortcut = x

        # First Conv layer
        x = layers.Conv2D(filters=filters, kernel_size=(3, 3), strides=(1, 1),␣
↪padding="same")(x)
        if self.batch_norm:
            x = layers.BatchNormalization()(x)
        x = layers.Activation(self.activation_func)(x)

        # Second Conv layer
        x = layers.Conv2D(filters=filters, kernel_size=(3, 3), strides=(1, 1),␣
↪padding="same")(x)
        if self.batch_norm:
            x = layers.BatchNormalization()(x)

        # Add the shortcut to the output
        if self.use_skip:
            x = tf.keras.layers.add([shortcut, x])
        x = layers.Activation(self.activation_func)(x)

        # Mean Pooling layer
        x = layers.AveragePooling2D(pool_size=(2, 2), strides=(2, 2),␣
↪padding="same")(x)

        # Dropout
        x = layers.Dropout(rate=self.drop_rate)(x)

        return x
```

```
  def build_cnn(self):
    input_tensor = layers.Input(shape=(self.height, self.width, self.depth))
    x = input_tensor

    for current_feature_maps in self.feature_maps:
      # Initial convolution
      x = layers.Conv2D(current_feature_maps, (3, 3), padding='same',␣
↪activation=None)(x)
      if self.batch_norm:
          x = layers.BatchNormalization()(x)
      x = layers.Activation(self.activation_func)(x)

      # Second convolution with skip connection
      shortcut = x
      x = layers.Conv2D(current_feature_maps, (3, 3), padding='same',␣
↪activation=None)(x)
      if self.batch_norm:
          x = layers.BatchNormalization()(x)
      if self.use_skip:
          x = layers.Add()([shortcut, x])
      x = layers.Activation(self.activation_func)(x)

      # Mean Pooling and Dropout
      x = layers.AveragePooling2D(pool_size=(2, 2), padding='same')(x)
      if self.drop_rate > 0:
          x = layers.Dropout(self.drop_rate)(x)

    x = layers.Flatten()(x)
    x = layers.Dense(self.num_classes, activation='softmax')(x)

    self.model = models.Model(inputs=input_tensor, outputs=x)
    self.model.compile(optimizer=self.optimizer,␣
↪loss='sparse_categorical_crossentropy', metrics=['accuracy'])

  def generate_mixup_data(self,x, y, alpha=0.2):
    """
    Apply mixup technique
    """
    if alpha > 0:
      lam = np.random.beta(alpha, alpha)
    else:
      lam = 1

    batch_size = x.shape[0]
    index = np.random.permutation(batch_size)
```

```
        mixed_x = lam * x + (1 - lam) * x[index]#mixup
        mixed_y = lam * y + (1 - lam) * y[index]#mixup

        return mixed_x, mixed_y



    def fit(self, data_manager, batch_size=None, num_epochs=None,␣
↪use_mixup=False, alpha=0.2):
        batch_size = self.batch_size if batch_size is None else batch_size
        num_epochs = self.num_epochs if num_epochs is None else num_epochs

        x_train_batch = self.optimize_data_pipeline(data_manager.ds_train,␣
↪batch_size=batch_size)
        x_val_batch = self.optimize_data_pipeline(data_manager.ds_val,␣
↪batch_size=batch_size)

        if self.is_augmentation:#code for data augmentation
          train_data_list, train_labels_list = [], []
          for data, labels in x_train_batch:
              train_data_list.append(data.numpy())
              train_labels_list.append(labels.numpy())

          train_data = np.concatenate(train_data_list, axis=0)
          train_labels = np.concatenate(train_labels_list, axis=0)

          datagen = ImageDataGenerator(rotation_range=10,
                                       width_shift_range=0.1,
                                       height_shift_range=0.1,
                                       horizontal_flip=True)

          datagen.fit(train_data)

          self.model.fit(datagen.flow(train_data, train_labels, batch_size=self.
↪batch_size),
                         validation_data=x_val_batch,
                         epochs=self.num_epochs,
                         verbose=self.verbose)

        if use_mixup:
          for epoch in range(num_epochs):
              for step, (x_batch, y_batch) in enumerate(x_train_batch):
                  # Apply mixup
                  mixed_x, mixed_y = self.generate_mixup_data(x_batch.numpy(),␣
↪y_batch.numpy(), alpha=alpha)

                  # Training step
```

```
                self.model.train_on_batch(mixed_x, mixed_y)
        else:
          self.model.fit(x_train_batch, validation_data=x_val_batch, epochs=self.
↪num_epochs, batch_size=self.batch_size, verbose=self.verbose)
```

[ ]:
```
model_3_7=YourModel(name='q3_7',
                     feature_maps=32,
                     num_classes=data_manager.n_classes,
                     num_blocks=4,
                     drop_rate=0.0,
                     batch_norm=True,
                     is_augmentation=True,
                     use_skip=True,
                     optimizer='sgd',
                     learning_rate=0.001)


model_3_7.build_cnn()
model_3_7.fit(data_manager,use_mixup=True)
```

```
Epoch 1/20
450/450 [==============================] - 9s 13ms/step - loss: 2.0255 -
accuracy: 0.2962 - val_loss: 2.0315 - val_accuracy: 0.3100
Epoch 2/20
450/450 [==============================] - 6s 13ms/step - loss: 1.6694 -
accuracy: 0.3902 - val_loss: 1.5157 - val_accuracy: 0.4380
Epoch 3/20
450/450 [==============================] - 6s 13ms/step - loss: 1.5102 -
accuracy: 0.4531 - val_loss: 2.1353 - val_accuracy: 0.3340
Epoch 4/20
450/450 [==============================] - 5s 12ms/step - loss: 1.4502 -
accuracy: 0.4640 - val_loss: 1.8327 - val_accuracy: 0.4120
Epoch 5/20
450/450 [==============================] - 6s 13ms/step - loss: 1.3412 -
accuracy: 0.5102 - val_loss: 1.9333 - val_accuracy: 0.3320
Epoch 6/20
450/450 [==============================] - 5s 11ms/step - loss: 1.2960 -
accuracy: 0.5253 - val_loss: 1.2722 - val_accuracy: 0.5240
Epoch 7/20
450/450 [==============================] - 6s 14ms/step - loss: 1.2365 -
accuracy: 0.5409 - val_loss: 1.7553 - val_accuracy: 0.4080
Epoch 8/20
450/450 [==============================] - 5s 12ms/step - loss: 1.1713 -
accuracy: 0.5664 - val_loss: 1.1427 - val_accuracy: 0.5820
Epoch 9/20
450/450 [==============================] - 7s 15ms/step - loss: 1.1278 -
accuracy: 0.5904 - val_loss: 1.2963 - val_accuracy: 0.5280
```

```
Epoch 10/20
450/450 [==============================] - 6s 13ms/step - loss: 1.1182 -
accuracy: 0.5798 - val_loss: 1.2589 - val_accuracy: 0.5900
Epoch 11/20
450/450 [==============================] - 5s 12ms/step - loss: 1.0554 -
accuracy: 0.6100 - val_loss: 1.2019 - val_accuracy: 0.5740
Epoch 12/20
450/450 [==============================] - 6s 13ms/step - loss: 1.0309 -
accuracy: 0.6213 - val_loss: 1.8021 - val_accuracy: 0.4760
Epoch 13/20
450/450 [==============================] - 5s 12ms/step - loss: 0.9816 -
accuracy: 0.6318 - val_loss: 1.5859 - val_accuracy: 0.4980
Epoch 14/20
450/450 [==============================] - 6s 14ms/step - loss: 0.9381 -
accuracy: 0.6631 - val_loss: 1.3570 - val_accuracy: 0.5500
Epoch 15/20
450/450 [==============================] - 5s 12ms/step - loss: 0.9351 -
accuracy: 0.6580 - val_loss: 1.3212 - val_accuracy: 0.5620
Epoch 16/20
450/450 [==============================] - 6s 14ms/step - loss: 0.8765 -
accuracy: 0.6804 - val_loss: 1.5741 - val_accuracy: 0.4840
Epoch 17/20
450/450 [==============================] - 5s 12ms/step - loss: 0.8558 -
accuracy: 0.6867 - val_loss: 1.2008 - val_accuracy: 0.5880
Epoch 18/20
450/450 [==============================] - 5s 12ms/step - loss: 0.8264 -
accuracy: 0.7022 - val_loss: 1.1152 - val_accuracy: 0.6000
Epoch 19/20
450/450 [==============================] - 6s 13ms/step - loss: 0.7997 -
accuracy: 0.7091 - val_loss: 1.2903 - val_accuracy: 0.5880
Epoch 20/20
450/450 [==============================] - 5s 12ms/step - loss: 0.7928 -
accuracy: 0.7131 - val_loss: 1.0881 - val_accuracy: 0.6220
```

Result: "loss: 0.7928 - accuracy: 0.7131 - val_loss: 1.0881 - val_accuracy: 0.6220"

It is noticed that val_accuracy improves.

[ ]:
```
#######The following code is just for further fine tuning the␣
↪parameters###########

# try to further fine tune the parameters to find the best combination
drop_rates = [0.0, 0.2, 0.5]
optimizers = ['sgd', 'adam', 'rmsprop']
alphas = [0.1, 0.2, 0.5]

# Placeholder
best_params = None
```

```python
best_val_accuracy = 0

# Iterate over all combinations
for drop_rate in drop_rates:
    for optimizer in optimizers:
        for alpha in alphas:
            # Create a model with the current parameter combination
            model_3_7_2 = YourModel(name='q3_7_2',
                                  feature_maps=32,
                                  num_classes=data_manager.n_classes,
                                  num_blocks=4,
                                  drop_rate=drop_rate,
                                  batch_norm=True,
                                  is_augmentation=True,
                                  use_skip=True,
                                  optimizer=optimizer,
                                  learning_rate=0.001)

            # Build and train the model
            model_3_7_2.build_cnn()
            model_3_7_2.fit(data_manager, use_mixup=True, alpha=alpha)

            # Evaluate the model on the validation set
            val_accuracy = max(model_3_7_2.model.history.
 ↪history['val_accuracy'])

            # Check if this performance is the best so far
            if val_accuracy> best_val_accuracy:
                best_val_performance = val_accuracy
                best_params = {'drop_rate': drop_rate, 'optimizer': optimizer,␣
 ↪'alpha': alpha}


print("Best Parameters:", best_params)
print("Best Validation Performance:", best_val_accuracy)
```

```
Epoch 1/20
450/450 [==============================] - 9s 13ms/step - loss: 2.0354 -
accuracy: 0.3029 - val_loss: 1.7770 - val_accuracy: 0.3340
Epoch 2/20
450/450 [==============================] - 6s 13ms/step - loss: 1.7012 -
accuracy: 0.3849 - val_loss: 1.5626 - val_accuracy: 0.4180
Epoch 3/20
450/450 [==============================] - 5s 12ms/step - loss: 1.5254 -
accuracy: 0.4424 - val_loss: 1.7653 - val_accuracy: 0.3640
Epoch 4/20
450/450 [==============================] - 6s 14ms/step - loss: 1.4283 -
```

```
accuracy: 0.4678 - val_loss: 1.5592 - val_accuracy: 0.4520
Epoch 5/20
450/450 [==============================] - 5s 12ms/step - loss: 1.3566 -
accuracy: 0.5080 - val_loss: 1.5697 - val_accuracy: 0.4660
Epoch 6/20
450/450 [==============================] - 6s 13ms/step - loss: 1.2920 -
accuracy: 0.5360 - val_loss: 1.3516 - val_accuracy: 0.5020
Epoch 7/20
450/450 [==============================] - 5s 12ms/step - loss: 1.2373 -
accuracy: 0.5447 - val_loss: 1.2674 - val_accuracy: 0.5360
Epoch 8/20
450/450 [==============================] - 6s 14ms/step - loss: 1.1924 -
accuracy: 0.5624 - val_loss: 1.5309 - val_accuracy: 0.4540
Epoch 9/20
450/450 [==============================] - 5s 12ms/step - loss: 1.1282 -
accuracy: 0.5951 - val_loss: 1.3653 - val_accuracy: 0.5280
Epoch 10/20
450/450 [==============================] - 6s 13ms/step - loss: 1.1126 -
accuracy: 0.5949 - val_loss: 1.7026 - val_accuracy: 0.4520
Epoch 11/20
450/450 [==============================] - 5s 12ms/step - loss: 1.0546 -
accuracy: 0.6198 - val_loss: 1.8052 - val_accuracy: 0.4560
Epoch 12/20
450/450 [==============================] - 5s 12ms/step - loss: 1.0238 -
accuracy: 0.6189 - val_loss: 1.4236 - val_accuracy: 0.5300
Epoch 13/20
450/450 [==============================] - 6s 14ms/step - loss: 0.9884 -
accuracy: 0.6373 - val_loss: 1.1623 - val_accuracy: 0.5960
Epoch 14/20
450/450 [==============================] - 5s 12ms/step - loss: 0.9452 -
accuracy: 0.6544 - val_loss: 1.4605 - val_accuracy: 0.5200
Epoch 15/20
450/450 [==============================] - 6s 14ms/step - loss: 0.9289 -
accuracy: 0.6598 - val_loss: 1.7715 - val_accuracy: 0.4540
Epoch 16/20
450/450 [==============================] - 6s 12ms/step - loss: 0.8675 -
accuracy: 0.6827 - val_loss: 1.1270 - val_accuracy: 0.6220
Epoch 17/20
450/450 [==============================] - 6s 13ms/step - loss: 0.8843 -
accuracy: 0.6780 - val_loss: 1.2103 - val_accuracy: 0.5900
Epoch 18/20
450/450 [==============================] - 5s 12ms/step - loss: 0.8475 -
accuracy: 0.6967 - val_loss: 2.5453 - val_accuracy: 0.3960
Epoch 19/20
450/450 [==============================] - 6s 14ms/step - loss: 0.7988 -
accuracy: 0.7047 - val_loss: 1.0827 - val_accuracy: 0.6320
Epoch 20/20
450/450 [==============================] - 5s 12ms/step - loss: 0.8029 -
```

```
accuracy: 0.7038 - val_loss: 1.4081 - val_accuracy: 0.5740
Epoch 1/20
450/450 [==============================] - 10s 17ms/step - loss: 2.0276 -
accuracy: 0.3073 - val_loss: 1.7634 - val_accuracy: 0.3740
Epoch 2/20
450/450 [==============================] - 5s 12ms/step - loss: 1.6939 -
accuracy: 0.3880 - val_loss: 1.6321 - val_accuracy: 0.4100
Epoch 3/20
450/450 [==============================] - 6s 14ms/step - loss: 1.5266 -
accuracy: 0.4431 - val_loss: 2.4063 - val_accuracy: 0.2760
Epoch 4/20
450/450 [==============================] - 5s 12ms/step - loss: 1.4411 -
accuracy: 0.4747 - val_loss: 1.5643 - val_accuracy: 0.4740
Epoch 5/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3557 -
accuracy: 0.4949 - val_loss: 1.4572 - val_accuracy: 0.5040
Epoch 6/20
450/450 [==============================] - 5s 12ms/step - loss: 1.3041 -
accuracy: 0.5211 - val_loss: 1.5791 - val_accuracy: 0.4600
Epoch 7/20
450/450 [==============================] - 6s 13ms/step - loss: 1.2475 -
accuracy: 0.5384 - val_loss: 1.4807 - val_accuracy: 0.4800
Epoch 8/20
450/450 [==============================] - 5s 12ms/step - loss: 1.2113 -
accuracy: 0.5629 - val_loss: 1.4128 - val_accuracy: 0.5000
Epoch 9/20
450/450 [==============================] - 6s 13ms/step - loss: 1.1169 -
accuracy: 0.5940 - val_loss: 1.1977 - val_accuracy: 0.5540
Epoch 10/20
450/450 [==============================] - 6s 12ms/step - loss: 1.1066 -
accuracy: 0.5933 - val_loss: 1.2769 - val_accuracy: 0.5420
Epoch 11/20
450/450 [==============================] - 5s 12ms/step - loss: 1.0626 -
accuracy: 0.6171 - val_loss: 1.8713 - val_accuracy: 0.4620
Epoch 12/20
450/450 [==============================] - 6s 13ms/step - loss: 1.0526 -
accuracy: 0.6167 - val_loss: 1.7249 - val_accuracy: 0.4420
Epoch 13/20
450/450 [==============================] - 5s 12ms/step - loss: 1.0185 -
accuracy: 0.6304 - val_loss: 1.3825 - val_accuracy: 0.5100
Epoch 14/20
450/450 [==============================] - 6s 14ms/step - loss: 0.9636 -
accuracy: 0.6531 - val_loss: 1.5294 - val_accuracy: 0.5140
Epoch 15/20
450/450 [==============================] - 5s 12ms/step - loss: 0.9518 -
accuracy: 0.6504 - val_loss: 1.2967 - val_accuracy: 0.5620
Epoch 16/20
450/450 [==============================] - 6s 14ms/step - loss: 0.9205 -
```

```
accuracy: 0.6691 - val_loss: 1.0866 - val_accuracy: 0.5940
Epoch 17/20
450/450 [==============================] - 5s 12ms/step - loss: 0.8674 -
accuracy: 0.6829 - val_loss: 1.2322 - val_accuracy: 0.5700
Epoch 18/20
450/450 [==============================] - 6s 14ms/step - loss: 0.8370 -
accuracy: 0.6951 - val_loss: 1.8385 - val_accuracy: 0.4560
Epoch 19/20
450/450 [==============================] - 5s 12ms/step - loss: 0.8487 -
accuracy: 0.6987 - val_loss: 1.3269 - val_accuracy: 0.5420
Epoch 20/20
450/450 [==============================] - 6s 13ms/step - loss: 0.7964 -
accuracy: 0.7082 - val_loss: 1.2248 - val_accuracy: 0.5980
Epoch 1/20
450/450 [==============================] - 8s 13ms/step - loss: 2.0375 -
accuracy: 0.3084 - val_loss: 2.2273 - val_accuracy: 0.3380
Epoch 2/20
450/450 [==============================] - 6s 14ms/step - loss: 1.6589 -
accuracy: 0.3953 - val_loss: 2.6954 - val_accuracy: 0.2200
Epoch 3/20
450/450 [==============================] - 5s 12ms/step - loss: 1.5299 -
accuracy: 0.4418 - val_loss: 1.4348 - val_accuracy: 0.4540
Epoch 4/20
450/450 [==============================] - 6s 14ms/step - loss: 1.4245 -
accuracy: 0.4831 - val_loss: 2.1491 - val_accuracy: 0.3300
Epoch 5/20
450/450 [==============================] - 5s 12ms/step - loss: 1.3310 -
accuracy: 0.5113 - val_loss: 1.8443 - val_accuracy: 0.4200
Epoch 6/20
450/450 [==============================] - 6s 14ms/step - loss: 1.2622 -
accuracy: 0.5358 - val_loss: 1.3673 - val_accuracy: 0.5080
Epoch 7/20
450/450 [==============================] - 5s 12ms/step - loss: 1.2205 -
accuracy: 0.5533 - val_loss: 1.6660 - val_accuracy: 0.4160
Epoch 8/20
450/450 [==============================] - 6s 14ms/step - loss: 1.1630 -
accuracy: 0.5733 - val_loss: 1.2215 - val_accuracy: 0.5580
Epoch 9/20
450/450 [==============================] - 5s 12ms/step - loss: 1.1237 -
accuracy: 0.5844 - val_loss: 1.1851 - val_accuracy: 0.5780
Epoch 10/20
450/450 [==============================] - 6s 14ms/step - loss: 1.1110 -
accuracy: 0.5929 - val_loss: 1.1893 - val_accuracy: 0.5720
Epoch 11/20
450/450 [==============================] - 5s 12ms/step - loss: 1.0519 -
accuracy: 0.6153 - val_loss: 1.5565 - val_accuracy: 0.4820
Epoch 12/20
450/450 [==============================] - 6s 13ms/step - loss: 1.0166 -
```

```
accuracy: 0.6329 - val_loss: 1.0919 - val_accuracy: 0.5980
Epoch 13/20
450/450 [==============================] - 6s 12ms/step - loss: 0.9756 -
accuracy: 0.6451 - val_loss: 1.6290 - val_accuracy: 0.4820
Epoch 14/20
450/450 [==============================] - 6s 14ms/step - loss: 0.9385 -
accuracy: 0.6647 - val_loss: 1.1805 - val_accuracy: 0.5720
Epoch 15/20
450/450 [==============================] - 5s 12ms/step - loss: 0.9199 -
accuracy: 0.6689 - val_loss: 1.1943 - val_accuracy: 0.5740
Epoch 16/20
450/450 [==============================] - 6s 14ms/step - loss: 0.8931 -
accuracy: 0.6767 - val_loss: 1.1112 - val_accuracy: 0.6000
Epoch 17/20
450/450 [==============================] - 5s 12ms/step - loss: 0.8654 -
accuracy: 0.6871 - val_loss: 1.1290 - val_accuracy: 0.6020
Epoch 18/20
450/450 [==============================] - 5s 12ms/step - loss: 0.8332 -
accuracy: 0.6969 - val_loss: 1.0585 - val_accuracy: 0.6140
Epoch 19/20
450/450 [==============================] - 6s 14ms/step - loss: 0.8083 -
accuracy: 0.7047 - val_loss: 1.5422 - val_accuracy: 0.5420
Epoch 20/20
450/450 [==============================] - 6s 13ms/step - loss: 0.7943 -
accuracy: 0.7098 - val_loss: 1.3984 - val_accuracy: 0.5540
Epoch 1/20
450/450 [==============================] - 13s 14ms/step - loss: 2.1137 -
accuracy: 0.2707 - val_loss: 1.7632 - val_accuracy: 0.3660
Epoch 2/20
450/450 [==============================] - 7s 15ms/step - loss: 1.7832 -
accuracy: 0.3436 - val_loss: 1.8600 - val_accuracy: 0.3700
Epoch 3/20
450/450 [==============================] - 6s 12ms/step - loss: 1.6610 -
accuracy: 0.3898 - val_loss: 1.5037 - val_accuracy: 0.4460
Epoch 4/20
450/450 [==============================] - 7s 14ms/step - loss: 1.5607 -
accuracy: 0.4240 - val_loss: 1.4796 - val_accuracy: 0.4440
Epoch 5/20
450/450 [==============================] - 6s 12ms/step - loss: 1.4926 -
accuracy: 0.4482 - val_loss: 1.4034 - val_accuracy: 0.4700
Epoch 6/20
450/450 [==============================] - 6s 14ms/step - loss: 1.4231 -
accuracy: 0.4744 - val_loss: 1.5006 - val_accuracy: 0.4240
Epoch 7/20
450/450 [==============================] - 6s 12ms/step - loss: 1.3639 -
accuracy: 0.4900 - val_loss: 1.5254 - val_accuracy: 0.4620
Epoch 8/20
450/450 [==============================] - 6s 14ms/step - loss: 1.2941 -
```

```
accuracy: 0.5158 - val_loss: 1.3231 - val_accuracy: 0.5100
Epoch 9/20
450/450 [==============================] - 6s 12ms/step - loss: 1.2272 -
accuracy: 0.5476 - val_loss: 1.6451 - val_accuracy: 0.4380
Epoch 10/20
450/450 [==============================] - 6s 14ms/step - loss: 1.1920 -
accuracy: 0.5569 - val_loss: 1.6471 - val_accuracy: 0.4220
Epoch 11/20
450/450 [==============================] - 5s 12ms/step - loss: 1.1457 -
accuracy: 0.5851 - val_loss: 1.8055 - val_accuracy: 0.3800
Epoch 12/20
450/450 [==============================] - 6s 14ms/step - loss: 1.1070 -
accuracy: 0.5951 - val_loss: 1.6724 - val_accuracy: 0.4480
Epoch 13/20
450/450 [==============================] - 6s 13ms/step - loss: 1.0424 -
accuracy: 0.6116 - val_loss: 1.0807 - val_accuracy: 0.6140
Epoch 14/20
450/450 [==============================] - 6s 13ms/step - loss: 1.0056 -
accuracy: 0.6293 - val_loss: 1.3152 - val_accuracy: 0.5120
Epoch 15/20
450/450 [==============================] - 5s 12ms/step - loss: 0.9842 -
accuracy: 0.6407 - val_loss: 1.1997 - val_accuracy: 0.5900
Epoch 16/20
450/450 [==============================] - 7s 14ms/step - loss: 0.9537 -
accuracy: 0.6460 - val_loss: 1.1192 - val_accuracy: 0.6100
Epoch 17/20
450/450 [==============================] - 6s 12ms/step - loss: 0.9025 -
accuracy: 0.6698 - val_loss: 1.5659 - val_accuracy: 0.4740
Epoch 18/20
450/450 [==============================] - 6s 14ms/step - loss: 0.8839 -
accuracy: 0.6818 - val_loss: 1.0431 - val_accuracy: 0.6540
Epoch 19/20
450/450 [==============================] - 6s 13ms/step - loss: 0.8552 -
accuracy: 0.6909 - val_loss: 1.5771 - val_accuracy: 0.4940
Epoch 20/20
450/450 [==============================] - 6s 14ms/step - loss: 0.8052 -
accuracy: 0.6998 - val_loss: 1.3468 - val_accuracy: 0.5360
Epoch 1/20
450/450 [==============================] - 13s 15ms/step - loss: 2.1242 -
accuracy: 0.2662 - val_loss: 2.4718 - val_accuracy: 0.2660
Epoch 2/20
450/450 [==============================] - 6s 14ms/step - loss: 1.7727 -
accuracy: 0.3456 - val_loss: 1.9867 - val_accuracy: 0.2840
Epoch 3/20
450/450 [==============================] - 6s 13ms/step - loss: 1.6398 -
accuracy: 0.3967 - val_loss: 1.7196 - val_accuracy: 0.4220
Epoch 4/20
450/450 [==============================] - 7s 14ms/step - loss: 1.5126 -
```

```
accuracy: 0.4433 - val_loss: 1.5613 - val_accuracy: 0.4380
Epoch 5/20
450/450 [==============================] - 6s 13ms/step - loss: 1.4180 -
accuracy: 0.4807 - val_loss: 1.3429 - val_accuracy: 0.5180
Epoch 6/20
450/450 [==============================] - 7s 15ms/step - loss: 1.3524 -
accuracy: 0.5022 - val_loss: 1.3083 - val_accuracy: 0.4980
Epoch 7/20
450/450 [==============================] - 6s 13ms/step - loss: 1.2924 -
accuracy: 0.5147 - val_loss: 1.4205 - val_accuracy: 0.4740
Epoch 8/20
450/450 [==============================] - 6s 12ms/step - loss: 1.2380 -
accuracy: 0.5331 - val_loss: 2.0379 - val_accuracy: 0.4020
Epoch 9/20
450/450 [==============================] - 7s 15ms/step - loss: 1.1939 -
accuracy: 0.5611 - val_loss: 1.1263 - val_accuracy: 0.6000
Epoch 10/20
450/450 [==============================] - 6s 13ms/step - loss: 1.1551 -
accuracy: 0.5733 - val_loss: 1.4301 - val_accuracy: 0.4740
Epoch 11/20
450/450 [==============================] - 7s 15ms/step - loss: 1.1111 -
accuracy: 0.5889 - val_loss: 1.2455 - val_accuracy: 0.5660
Epoch 12/20
450/450 [==============================] - 6s 13ms/step - loss: 1.0774 -
accuracy: 0.6004 - val_loss: 1.3632 - val_accuracy: 0.5100
Epoch 13/20
450/450 [==============================] - 7s 15ms/step - loss: 1.0208 -
accuracy: 0.6198 - val_loss: 1.2225 - val_accuracy: 0.5660
Epoch 14/20
450/450 [==============================] - 6s 12ms/step - loss: 0.9744 -
accuracy: 0.6424 - val_loss: 1.6167 - val_accuracy: 0.5060
Epoch 15/20
450/450 [==============================] - 7s 14ms/step - loss: 0.9561 -
accuracy: 0.6511 - val_loss: 1.0791 - val_accuracy: 0.6220
Epoch 16/20
450/450 [==============================] - 6s 13ms/step - loss: 0.9172 -
accuracy: 0.6627 - val_loss: 1.0447 - val_accuracy: 0.6240
Epoch 17/20
450/450 [==============================] - 6s 14ms/step - loss: 0.8756 -
accuracy: 0.6791 - val_loss: 1.1586 - val_accuracy: 0.6160
Epoch 18/20
450/450 [==============================] - 6s 12ms/step - loss: 0.8612 -
accuracy: 0.6871 - val_loss: 1.1174 - val_accuracy: 0.6120
Epoch 19/20
450/450 [==============================] - 7s 15ms/step - loss: 0.8359 -
accuracy: 0.6918 - val_loss: 1.8062 - val_accuracy: 0.4760
Epoch 20/20
450/450 [==============================] - 6s 12ms/step - loss: 0.7982 -
```

```
accuracy: 0.7031 - val_loss: 1.2887 - val_accuracy: 0.5780
Epoch 1/20
450/450 [==============================] - 13s 14ms/step - loss: 2.1532 -
accuracy: 0.2756 - val_loss: 1.9820 - val_accuracy: 0.2940
Epoch 2/20
450/450 [==============================] - 7s 15ms/step - loss: 1.7762 -
accuracy: 0.3469 - val_loss: 2.4789 - val_accuracy: 0.2040
Epoch 3/20
450/450 [==============================] - 5s 12ms/step - loss: 1.6531 -
accuracy: 0.3898 - val_loss: 2.7400 - val_accuracy: 0.2960
Epoch 4/20
450/450 [==============================] - 6s 14ms/step - loss: 1.5396 -
accuracy: 0.4251 - val_loss: 1.9080 - val_accuracy: 0.3280
Epoch 5/20
450/450 [==============================] - 6s 13ms/step - loss: 1.4304 -
accuracy: 0.4662 - val_loss: 1.6940 - val_accuracy: 0.3920
Epoch 6/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3760 -
accuracy: 0.4813 - val_loss: 3.2806 - val_accuracy: 0.2300
Epoch 7/20
450/450 [==============================] - 6s 12ms/step - loss: 1.3063 -
accuracy: 0.5227 - val_loss: 1.4182 - val_accuracy: 0.4960
Epoch 8/20
450/450 [==============================] - 6s 14ms/step - loss: 1.2339 -
accuracy: 0.5373 - val_loss: 1.6909 - val_accuracy: 0.4320
Epoch 9/20
450/450 [==============================] - 6s 12ms/step - loss: 1.1874 -
accuracy: 0.5573 - val_loss: 1.4353 - val_accuracy: 0.5320
Epoch 10/20
450/450 [==============================] - 7s 14ms/step - loss: 1.1393 -
accuracy: 0.5753 - val_loss: 1.3127 - val_accuracy: 0.5220
Epoch 11/20
450/450 [==============================] - 6s 12ms/step - loss: 1.1001 -
accuracy: 0.5980 - val_loss: 1.1823 - val_accuracy: 0.5380
Epoch 12/20
450/450 [==============================] - 7s 14ms/step - loss: 1.0573 -
accuracy: 0.6102 - val_loss: 1.2395 - val_accuracy: 0.5720
Epoch 13/20
450/450 [==============================] - 5s 12ms/step - loss: 1.0240 -
accuracy: 0.6204 - val_loss: 1.7087 - val_accuracy: 0.4740
Epoch 14/20
450/450 [==============================] - 6s 14ms/step - loss: 0.9672 -
accuracy: 0.6433 - val_loss: 1.1337 - val_accuracy: 0.6340
Epoch 15/20
450/450 [==============================] - 6s 12ms/step - loss: 0.9508 -
accuracy: 0.6522 - val_loss: 1.7439 - val_accuracy: 0.4740
Epoch 16/20
450/450 [==============================] - 7s 15ms/step - loss: 0.9058 -
```

```
accuracy: 0.6656 - val_loss: 1.1081 - val_accuracy: 0.5900
Epoch 17/20
450/450 [==============================] - 6s 12ms/step - loss: 0.8605 -
accuracy: 0.6789 - val_loss: 1.2743 - val_accuracy: 0.5580
Epoch 18/20
450/450 [==============================] - 7s 15ms/step - loss: 0.8406 -
accuracy: 0.6891 - val_loss: 1.1196 - val_accuracy: 0.6180
Epoch 19/20
450/450 [==============================] - 6s 12ms/step - loss: 0.8105 -
accuracy: 0.7016 - val_loss: 1.2957 - val_accuracy: 0.5540
Epoch 20/20
450/450 [==============================] - 6s 14ms/step - loss: 0.7908 -
accuracy: 0.7016 - val_loss: 1.3962 - val_accuracy: 0.5580
Epoch 1/20
450/450 [==============================] - 11s 14ms/step - loss: 2.2347 -
accuracy: 0.2516 - val_loss: 2.2594 - val_accuracy: 0.2440
Epoch 2/20
450/450 [==============================] - 5s 12ms/step - loss: 1.8123 -
accuracy: 0.3407 - val_loss: 1.6445 - val_accuracy: 0.3840
Epoch 3/20
450/450 [==============================] - 6s 13ms/step - loss: 1.6761 -
accuracy: 0.3796 - val_loss: 1.9524 - val_accuracy: 0.3660
Epoch 4/20
450/450 [==============================] - 5s 12ms/step - loss: 1.5618 -
accuracy: 0.4220 - val_loss: 1.5073 - val_accuracy: 0.4420
Epoch 5/20
450/450 [==============================] - 5s 12ms/step - loss: 1.4692 -
accuracy: 0.4542 - val_loss: 2.3699 - val_accuracy: 0.3000
Epoch 6/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3974 -
accuracy: 0.4800 - val_loss: 2.2974 - val_accuracy: 0.3520
Epoch 7/20
450/450 [==============================] - 5s 12ms/step - loss: 1.3239 -
accuracy: 0.5131 - val_loss: 2.1349 - val_accuracy: 0.3820
Epoch 8/20
450/450 [==============================] - 6s 14ms/step - loss: 1.2688 -
accuracy: 0.5322 - val_loss: 1.4154 - val_accuracy: 0.5040
Epoch 9/20
450/450 [==============================] - 5s 12ms/step - loss: 1.2362 -
accuracy: 0.5371 - val_loss: 1.6561 - val_accuracy: 0.4480
Epoch 10/20
450/450 [==============================] - 6s 14ms/step - loss: 1.1827 -
accuracy: 0.5582 - val_loss: 1.7021 - val_accuracy: 0.4480
Epoch 11/20
450/450 [==============================] - 5s 12ms/step - loss: 1.1562 -
accuracy: 0.5731 - val_loss: 1.6370 - val_accuracy: 0.4780
Epoch 12/20
450/450 [==============================] - 5s 12ms/step - loss: 1.1095 -
```

```
accuracy: 0.5922 - val_loss: 2.7953 - val_accuracy: 0.3160
Epoch 13/20
450/450 [==============================] - 6s 14ms/step - loss: 1.0570 -
accuracy: 0.6129 - val_loss: 1.7767 - val_accuracy: 0.4960
Epoch 14/20
450/450 [==============================] - 5s 12ms/step - loss: 1.0465 -
accuracy: 0.6189 - val_loss: 1.9187 - val_accuracy: 0.4540
Epoch 15/20
450/450 [==============================] - 6s 13ms/step - loss: 1.0199 -
accuracy: 0.6251 - val_loss: 1.0610 - val_accuracy: 0.6340
Epoch 16/20
450/450 [==============================] - 6s 12ms/step - loss: 0.9839 -
accuracy: 0.6367 - val_loss: 1.4212 - val_accuracy: 0.5380
Epoch 17/20
450/450 [==============================] - 6s 13ms/step - loss: 0.9423 -
accuracy: 0.6560 - val_loss: 1.1197 - val_accuracy: 0.6260
Epoch 18/20
450/450 [==============================] - 6s 14ms/step - loss: 0.9156 -
accuracy: 0.6678 - val_loss: 1.3959 - val_accuracy: 0.5300
Epoch 19/20
450/450 [==============================] - 5s 12ms/step - loss: 0.8951 -
accuracy: 0.6720 - val_loss: 1.1148 - val_accuracy: 0.6180
Epoch 20/20
450/450 [==============================] - 6s 14ms/step - loss: 0.8747 -
accuracy: 0.6816 - val_loss: 1.0702 - val_accuracy: 0.6260
Epoch 1/20
450/450 [==============================] - 11s 14ms/step - loss: 2.2262 -
accuracy: 0.2522 - val_loss: 1.9462 - val_accuracy: 0.2900
Epoch 2/20
450/450 [==============================] - 5s 12ms/step - loss: 1.8246 -
accuracy: 0.3378 - val_loss: 1.6210 - val_accuracy: 0.4000
Epoch 3/20
450/450 [==============================] - 6s 14ms/step - loss: 1.6568 -
accuracy: 0.3898 - val_loss: 1.5594 - val_accuracy: 0.4180
Epoch 4/20
450/450 [==============================] - 5s 12ms/step - loss: 1.5673 -
accuracy: 0.4262 - val_loss: 2.2974 - val_accuracy: 0.2980
Epoch 5/20
450/450 [==============================] - 6s 14ms/step - loss: 1.4817 -
accuracy: 0.4593 - val_loss: 1.3954 - val_accuracy: 0.4820
Epoch 6/20
450/450 [==============================] - 5s 12ms/step - loss: 1.4050 -
accuracy: 0.4771 - val_loss: 1.4313 - val_accuracy: 0.5060
Epoch 7/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3603 -
accuracy: 0.4942 - val_loss: 1.5945 - val_accuracy: 0.4440
Epoch 8/20
450/450 [==============================] - 5s 12ms/step - loss: 1.2967 -
```

```
accuracy: 0.5213 - val_loss: 1.2634 - val_accuracy: 0.5360
Epoch 9/20
450/450 [==============================] - 5s 12ms/step - loss: 1.2393 -
accuracy: 0.5373 - val_loss: 1.4708 - val_accuracy: 0.4800
Epoch 10/20
450/450 [==============================] - 6s 14ms/step - loss: 1.1984 -
accuracy: 0.5513 - val_loss: 1.3789 - val_accuracy: 0.5180
Epoch 11/20
450/450 [==============================] - 5s 12ms/step - loss: 1.1469 -
accuracy: 0.5780 - val_loss: 1.1787 - val_accuracy: 0.5680
Epoch 12/20
450/450 [==============================] - 6s 13ms/step - loss: 1.1252 -
accuracy: 0.5791 - val_loss: 1.4794 - val_accuracy: 0.5060
Epoch 13/20
450/450 [==============================] - 5s 12ms/step - loss: 1.0845 -
accuracy: 0.6033 - val_loss: 1.5160 - val_accuracy: 0.4860
Epoch 14/20
450/450 [==============================] - 6s 14ms/step - loss: 1.0525 -
accuracy: 0.6142 - val_loss: 1.3910 - val_accuracy: 0.5440
Epoch 15/20
450/450 [==============================] - 5s 12ms/step - loss: 1.0242 -
accuracy: 0.6160 - val_loss: 1.7600 - val_accuracy: 0.4880
Epoch 16/20
450/450 [==============================] - 5s 12ms/step - loss: 0.9876 -
accuracy: 0.6482 - val_loss: 1.2783 - val_accuracy: 0.5620
Epoch 17/20
450/450 [==============================] - 6s 14ms/step - loss: 0.9623 -
accuracy: 0.6491 - val_loss: 1.6105 - val_accuracy: 0.4980
Epoch 18/20
450/450 [==============================] - 5s 12ms/step - loss: 0.9393 -
accuracy: 0.6609 - val_loss: 1.4535 - val_accuracy: 0.5200
Epoch 19/20
450/450 [==============================] - 6s 14ms/step - loss: 0.9060 -
accuracy: 0.6744 - val_loss: 1.1326 - val_accuracy: 0.6240
Epoch 20/20
450/450 [==============================] - 5s 12ms/step - loss: 0.8991 -
accuracy: 0.6704 - val_loss: 1.1843 - val_accuracy: 0.6280
Epoch 1/20
450/450 [==============================] - 10s 13ms/step - loss: 2.2104 -
accuracy: 0.2636 - val_loss: 2.5101 - val_accuracy: 0.2780
Epoch 2/20
450/450 [==============================] - 6s 14ms/step - loss: 1.7879 -
accuracy: 0.3387 - val_loss: 2.6973 - val_accuracy: 0.2820
Epoch 3/20
450/450 [==============================] - 5s 12ms/step - loss: 1.6782 -
accuracy: 0.3733 - val_loss: 3.3359 - val_accuracy: 0.1840
Epoch 4/20
450/450 [==============================] - 6s 14ms/step - loss: 1.5512 -
```

```
accuracy: 0.4251 - val_loss: 1.4515 - val_accuracy: 0.4600
Epoch 5/20
450/450 [==============================] - 6s 13ms/step - loss: 1.4819 -
accuracy: 0.4424 - val_loss: 1.8120 - val_accuracy: 0.3900
Epoch 6/20
450/450 [==============================] - 6s 13ms/step - loss: 1.4198 -
accuracy: 0.4776 - val_loss: 2.3389 - val_accuracy: 0.3000
Epoch 7/20
450/450 [==============================] - 6s 13ms/step - loss: 1.3391 -
accuracy: 0.5049 - val_loss: 1.3621 - val_accuracy: 0.5120
Epoch 8/20
450/450 [==============================] - 6s 13ms/step - loss: 1.2726 -
accuracy: 0.5269 - val_loss: 1.2940 - val_accuracy: 0.5440
Epoch 9/20
450/450 [==============================] - 5s 12ms/step - loss: 1.2359 -
accuracy: 0.5476 - val_loss: 1.8727 - val_accuracy: 0.4100
Epoch 10/20
450/450 [==============================] - 6s 14ms/step - loss: 1.1881 -
accuracy: 0.5513 - val_loss: 1.2728 - val_accuracy: 0.5100
Epoch 11/20
450/450 [==============================] - 5s 12ms/step - loss: 1.1469 -
accuracy: 0.5804 - val_loss: 1.2281 - val_accuracy: 0.5520
Epoch 12/20
450/450 [==============================] - 6s 14ms/step - loss: 1.1189 -
accuracy: 0.5813 - val_loss: 1.4507 - val_accuracy: 0.5080
Epoch 13/20
450/450 [==============================] - 5s 12ms/step - loss: 1.0768 -
accuracy: 0.6100 - val_loss: 1.0950 - val_accuracy: 0.6020
Epoch 14/20
450/450 [==============================] - 5s 12ms/step - loss: 1.0618 -
accuracy: 0.6129 - val_loss: 1.4317 - val_accuracy: 0.5660
Epoch 15/20
450/450 [==============================] - 5s 12ms/step - loss: 1.0129 -
accuracy: 0.6316 - val_loss: 1.0892 - val_accuracy: 0.6300
Epoch 16/20
450/450 [==============================] - 6s 12ms/step - loss: 1.0033 -
accuracy: 0.6300 - val_loss: 1.2042 - val_accuracy: 0.5900
Epoch 17/20
450/450 [==============================] - 6s 12ms/step - loss: 0.9634 -
accuracy: 0.6509 - val_loss: 1.8299 - val_accuracy: 0.4560
Epoch 18/20
450/450 [==============================] - 6s 13ms/step - loss: 0.9452 -
accuracy: 0.6607 - val_loss: 1.1356 - val_accuracy: 0.6120
Epoch 19/20
450/450 [==============================] - 5s 12ms/step - loss: 0.9272 -
accuracy: 0.6589 - val_loss: 1.3072 - val_accuracy: 0.5800
Epoch 20/20
450/450 [==============================] - 6s 14ms/step - loss: 0.9116 -
```

```
accuracy: 0.6673 - val_loss: 1.0930 - val_accuracy: 0.6180
Epoch 1/20
450/450 [==============================] - 9s 15ms/step - loss: 2.1962 -
accuracy: 0.2596 - val_loss: 1.7909 - val_accuracy: 0.2980
Epoch 2/20
450/450 [==============================] - 6s 13ms/step - loss: 1.8666 -
accuracy: 0.3247 - val_loss: 2.1943 - val_accuracy: 0.2660
Epoch 3/20
450/450 [==============================] - 6s 14ms/step - loss: 1.7486 -
accuracy: 0.3536 - val_loss: 2.0690 - val_accuracy: 0.3300
Epoch 4/20
450/450 [==============================] - 6s 14ms/step - loss: 1.6602 -
accuracy: 0.3878 - val_loss: 1.9508 - val_accuracy: 0.3540
Epoch 5/20
450/450 [==============================] - 6s 13ms/step - loss: 1.5763 -
accuracy: 0.4138 - val_loss: 1.5519 - val_accuracy: 0.4360
Epoch 6/20
450/450 [==============================] - 7s 15ms/step - loss: 1.5061 -
accuracy: 0.4449 - val_loss: 2.3199 - val_accuracy: 0.3520
Epoch 7/20
450/450 [==============================] - 6s 12ms/step - loss: 1.5075 -
accuracy: 0.4413 - val_loss: 1.6965 - val_accuracy: 0.4120
Epoch 8/20
450/450 [==============================] - 7s 14ms/step - loss: 1.4493 -
accuracy: 0.4682 - val_loss: 1.7184 - val_accuracy: 0.4420
Epoch 9/20
450/450 [==============================] - 6s 12ms/step - loss: 1.4069 -
accuracy: 0.4771 - val_loss: 1.4615 - val_accuracy: 0.4820
Epoch 10/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3738 -
accuracy: 0.4898 - val_loss: 1.3040 - val_accuracy: 0.5360
Epoch 11/20
450/450 [==============================] - 6s 13ms/step - loss: 1.3305 -
accuracy: 0.5013 - val_loss: 2.4997 - val_accuracy: 0.3680
Epoch 12/20
450/450 [==============================] - 7s 15ms/step - loss: 1.3334 -
accuracy: 0.5049 - val_loss: 1.4311 - val_accuracy: 0.4920
Epoch 13/20
450/450 [==============================] - 6s 13ms/step - loss: 1.2870 -
accuracy: 0.5231 - val_loss: 1.3582 - val_accuracy: 0.5260
Epoch 14/20
450/450 [==============================] - 6s 14ms/step - loss: 1.2899 -
accuracy: 0.5278 - val_loss: 1.2296 - val_accuracy: 0.5580
Epoch 15/20
450/450 [==============================] - 6s 13ms/step - loss: 1.2610 -
accuracy: 0.5402 - val_loss: 1.4789 - val_accuracy: 0.5040
Epoch 16/20
450/450 [==============================] - 6s 14ms/step - loss: 1.2314 -
```

```
accuracy: 0.5507 - val_loss: 1.2562 - val_accuracy: 0.5580
Epoch 17/20
450/450 [==============================] - 6s 12ms/step - loss: 1.1846 -
accuracy: 0.5562 - val_loss: 1.2892 - val_accuracy: 0.5440
Epoch 18/20
450/450 [==============================] - 6s 13ms/step - loss: 1.1665 -
accuracy: 0.5731 - val_loss: 2.1407 - val_accuracy: 0.3940
Epoch 19/20
450/450 [==============================] - 7s 15ms/step - loss: 1.1645 -
accuracy: 0.5762 - val_loss: 1.2523 - val_accuracy: 0.5460
Epoch 20/20
450/450 [==============================] - 6s 13ms/step - loss: 1.1485 -
accuracy: 0.5840 - val_loss: 1.5985 - val_accuracy: 0.4920
Epoch 1/20
450/450 [==============================] - 9s 14ms/step - loss: 2.1464 -
accuracy: 0.2867 - val_loss: 2.1429 - val_accuracy: 0.2520
Epoch 2/20
450/450 [==============================] - 7s 15ms/step - loss: 1.8704 -
accuracy: 0.3380 - val_loss: 2.2154 - val_accuracy: 0.2780
Epoch 3/20
450/450 [==============================] - 6s 12ms/step - loss: 1.6835 -
accuracy: 0.3876 - val_loss: 1.5998 - val_accuracy: 0.4120
Epoch 4/20
450/450 [==============================] - 7s 15ms/step - loss: 1.5907 -
accuracy: 0.4142 - val_loss: 1.6238 - val_accuracy: 0.4380
Epoch 5/20
450/450 [==============================] - 6s 12ms/step - loss: 1.5617 -
accuracy: 0.4278 - val_loss: 1.6735 - val_accuracy: 0.4300
Epoch 6/20
450/450 [==============================] - 7s 15ms/step - loss: 1.4707 -
accuracy: 0.4636 - val_loss: 1.3083 - val_accuracy: 0.5240
Epoch 7/20
450/450 [==============================] - 6s 12ms/step - loss: 1.4709 -
accuracy: 0.4658 - val_loss: 1.2445 - val_accuracy: 0.5180
Epoch 8/20
450/450 [==============================] - 7s 14ms/step - loss: 1.4039 -
accuracy: 0.4838 - val_loss: 1.2951 - val_accuracy: 0.5440
Epoch 9/20
450/450 [==============================] - 6s 12ms/step - loss: 1.3799 -
accuracy: 0.4860 - val_loss: 1.5167 - val_accuracy: 0.4740
Epoch 10/20
450/450 [==============================] - 6s 12ms/step - loss: 1.3333 -
accuracy: 0.5098 - val_loss: 1.4785 - val_accuracy: 0.4700
Epoch 11/20
450/450 [==============================] - 7s 14ms/step - loss: 1.2915 -
accuracy: 0.5336 - val_loss: 1.4632 - val_accuracy: 0.4940
Epoch 12/20
450/450 [==============================] - 6s 12ms/step - loss: 1.2998 -
```

```
accuracy: 0.5204 - val_loss: 1.3469 - val_accuracy: 0.5280
Epoch 13/20
450/450 [==============================] - 6s 14ms/step - loss: 1.2550 -
accuracy: 0.5376 - val_loss: 1.3886 - val_accuracy: 0.5200
Epoch 14/20
450/450 [==============================] - 6s 13ms/step - loss: 1.2293 -
accuracy: 0.5520 - val_loss: 1.4196 - val_accuracy: 0.5020
Epoch 15/20
450/450 [==============================] - 7s 15ms/step - loss: 1.2298 -
accuracy: 0.5513 - val_loss: 1.6946 - val_accuracy: 0.4480
Epoch 16/20
450/450 [==============================] - 6s 13ms/step - loss: 1.2066 -
accuracy: 0.5522 - val_loss: 1.4682 - val_accuracy: 0.5100
Epoch 17/20
450/450 [==============================] - 6s 14ms/step - loss: 1.1609 -
accuracy: 0.5667 - val_loss: 1.7355 - val_accuracy: 0.4380
Epoch 18/20
450/450 [==============================] - 6s 12ms/step - loss: 1.1650 -
accuracy: 0.5718 - val_loss: 1.2265 - val_accuracy: 0.5700
Epoch 19/20
450/450 [==============================] - 6s 14ms/step - loss: 1.1173 -
accuracy: 0.5989 - val_loss: 1.2678 - val_accuracy: 0.5460
Epoch 20/20
450/450 [==============================] - 6s 12ms/step - loss: 1.1475 -
accuracy: 0.5780 - val_loss: 1.3032 - val_accuracy: 0.5600
Epoch 1/20
450/450 [==============================] - 10s 14ms/step - loss: 2.1304 -
accuracy: 0.2751 - val_loss: 1.9710 - val_accuracy: 0.3080
Epoch 2/20
450/450 [==============================] - 6s 14ms/step - loss: 1.8220 -
accuracy: 0.3484 - val_loss: 2.5522 - val_accuracy: 0.2420
Epoch 3/20
450/450 [==============================] - 6s 12ms/step - loss: 1.7116 -
accuracy: 0.3884 - val_loss: 1.6028 - val_accuracy: 0.4020
Epoch 4/20
450/450 [==============================] - 6s 14ms/step - loss: 1.6057 -
accuracy: 0.4200 - val_loss: 1.5387 - val_accuracy: 0.4840
Epoch 5/20
450/450 [==============================] - 6s 13ms/step - loss: 1.5308 -
accuracy: 0.4329 - val_loss: 1.4979 - val_accuracy: 0.4260
Epoch 6/20
450/450 [==============================] - 6s 14ms/step - loss: 1.4641 -
accuracy: 0.4656 - val_loss: 1.4542 - val_accuracy: 0.4740
Epoch 7/20
450/450 [==============================] - 6s 13ms/step - loss: 1.4211 -
accuracy: 0.4816 - val_loss: 1.6783 - val_accuracy: 0.4500
Epoch 8/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3972 -
```

```
accuracy: 0.4920 - val_loss: 1.3543 - val_accuracy: 0.5020
Epoch 9/20
450/450 [==============================] - 6s 13ms/step - loss: 1.3387 -
accuracy: 0.5093 - val_loss: 1.2256 - val_accuracy: 0.5600
Epoch 10/20
450/450 [==============================] - 7s 15ms/step - loss: 1.3237 -
accuracy: 0.5142 - val_loss: 1.4115 - val_accuracy: 0.5120
Epoch 11/20
450/450 [==============================] - 6s 12ms/step - loss: 1.2907 -
accuracy: 0.5240 - val_loss: 1.3837 - val_accuracy: 0.5180
Epoch 12/20
450/450 [==============================] - 7s 14ms/step - loss: 1.2733 -
accuracy: 0.5358 - val_loss: 1.3216 - val_accuracy: 0.5440
Epoch 13/20
450/450 [==============================] - 7s 15ms/step - loss: 1.2231 -
accuracy: 0.5484 - val_loss: 1.1485 - val_accuracy: 0.5740
Epoch 14/20
450/450 [==============================] - 6s 13ms/step - loss: 1.2128 -
accuracy: 0.5538 - val_loss: 1.2171 - val_accuracy: 0.5580
Epoch 15/20
450/450 [==============================] - 7s 15ms/step - loss: 1.1891 -
accuracy: 0.5680 - val_loss: 1.2815 - val_accuracy: 0.5480
Epoch 16/20
450/450 [==============================] - 6s 12ms/step - loss: 1.1667 -
accuracy: 0.5658 - val_loss: 1.1443 - val_accuracy: 0.5980
Epoch 17/20
450/450 [==============================] - 7s 15ms/step - loss: 1.1310 -
accuracy: 0.5802 - val_loss: 1.2311 - val_accuracy: 0.5840
Epoch 18/20
450/450 [==============================] - 6s 13ms/step - loss: 1.1186 -
accuracy: 0.5900 - val_loss: 1.5499 - val_accuracy: 0.4860
Epoch 19/20
450/450 [==============================] - 6s 14ms/step - loss: 1.1129 -
accuracy: 0.5904 - val_loss: 1.4807 - val_accuracy: 0.5280
Epoch 20/20
450/450 [==============================] - 6s 13ms/step - loss: 1.0809 -
accuracy: 0.6058 - val_loss: 1.3984 - val_accuracy: 0.5300
Epoch 1/20
450/450 [==============================] - 13s 14ms/step - loss: 2.2635 -
accuracy: 0.2509 - val_loss: 1.8797 - val_accuracy: 0.2500
Epoch 2/20
450/450 [==============================] - 7s 16ms/step - loss: 1.8533 -
accuracy: 0.3260 - val_loss: 2.0486 - val_accuracy: 0.3200
Epoch 3/20
450/450 [==============================] - 6s 14ms/step - loss: 1.6932 -
accuracy: 0.3756 - val_loss: 2.0868 - val_accuracy: 0.2900
Epoch 4/20
450/450 [==============================] - 7s 15ms/step - loss: 1.5910 -
```

```
accuracy: 0.4127 - val_loss: 1.4830 - val_accuracy: 0.4420
Epoch 5/20
450/450 [==============================] - 6s 13ms/step - loss: 1.4967 -
accuracy: 0.4440 - val_loss: 1.6783 - val_accuracy: 0.3780
Epoch 6/20
450/450 [==============================] - 7s 15ms/step - loss: 1.4337 -
accuracy: 0.4649 - val_loss: 1.7998 - val_accuracy: 0.3320
Epoch 7/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3942 -
accuracy: 0.4798 - val_loss: 2.0606 - val_accuracy: 0.3800
Epoch 8/20
450/450 [==============================] - 7s 15ms/step - loss: 1.3491 -
accuracy: 0.4913 - val_loss: 4.2999 - val_accuracy: 0.1740
Epoch 9/20
450/450 [==============================] - 7s 15ms/step - loss: 1.2911 -
accuracy: 0.5178 - val_loss: 1.4714 - val_accuracy: 0.4640
Epoch 10/20
450/450 [==============================] - 6s 13ms/step - loss: 1.2628 -
accuracy: 0.5287 - val_loss: 1.1700 - val_accuracy: 0.5700
Epoch 11/20
450/450 [==============================] - 7s 15ms/step - loss: 1.2219 -
accuracy: 0.5469 - val_loss: 1.4337 - val_accuracy: 0.4740
Epoch 12/20
450/450 [==============================] - 7s 15ms/step - loss: 1.2024 -
accuracy: 0.5553 - val_loss: 1.5398 - val_accuracy: 0.4780
Epoch 13/20
450/450 [==============================] - 6s 13ms/step - loss: 1.1542 -
accuracy: 0.5718 - val_loss: 1.1351 - val_accuracy: 0.5760
Epoch 14/20
450/450 [==============================] - 7s 16ms/step - loss: 1.1335 -
accuracy: 0.5811 - val_loss: 1.2475 - val_accuracy: 0.5400
Epoch 15/20
450/450 [==============================] - 6s 13ms/step - loss: 1.1182 -
accuracy: 0.5824 - val_loss: 1.1590 - val_accuracy: 0.5940
Epoch 16/20
450/450 [==============================] - 7s 15ms/step - loss: 1.0863 -
accuracy: 0.6042 - val_loss: 1.1417 - val_accuracy: 0.5900
Epoch 17/20
450/450 [==============================] - 6s 13ms/step - loss: 1.0533 -
accuracy: 0.6127 - val_loss: 1.2680 - val_accuracy: 0.5820
Epoch 18/20
450/450 [==============================] - 7s 15ms/step - loss: 1.0240 -
accuracy: 0.6173 - val_loss: 1.3053 - val_accuracy: 0.5560
Epoch 19/20
450/450 [==============================] - 6s 14ms/step - loss: 1.0034 -
accuracy: 0.6351 - val_loss: 1.2813 - val_accuracy: 0.5240
Epoch 20/20
450/450 [==============================] - 7s 16ms/step - loss: 0.9900 -
```

```
accuracy: 0.6449 - val_loss: 1.1672 - val_accuracy: 0.5900
Epoch 1/20
450/450 [==============================] - 13s 16ms/step - loss: 2.2121 -
accuracy: 0.2640 - val_loss: 1.7624 - val_accuracy: 0.3340
Epoch 2/20
450/450 [==============================] - 6s 13ms/step - loss: 1.8658 -
accuracy: 0.3269 - val_loss: 2.2417 - val_accuracy: 0.2860
Epoch 3/20
450/450 [==============================] - 6s 14ms/step - loss: 1.7258 -
accuracy: 0.3711 - val_loss: 1.4517 - val_accuracy: 0.4540
Epoch 4/20
450/450 [==============================] - 7s 16ms/step - loss: 1.6088 -
accuracy: 0.4007 - val_loss: 2.0467 - val_accuracy: 0.3020
Epoch 5/20
450/450 [==============================] - 6s 14ms/step - loss: 1.5363 -
accuracy: 0.4218 - val_loss: 1.5731 - val_accuracy: 0.4240
Epoch 6/20
450/450 [==============================] - 7s 15ms/step - loss: 1.4784 -
accuracy: 0.4540 - val_loss: 1.4232 - val_accuracy: 0.4700
Epoch 7/20
450/450 [==============================] - 6s 14ms/step - loss: 1.4235 -
accuracy: 0.4738 - val_loss: 1.6543 - val_accuracy: 0.4120
Epoch 8/20
450/450 [==============================] - 7s 16ms/step - loss: 1.3700 -
accuracy: 0.4902 - val_loss: 1.1865 - val_accuracy: 0.5740
Epoch 9/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3058 -
accuracy: 0.5129 - val_loss: 1.7424 - val_accuracy: 0.3800
Epoch 10/20
450/450 [==============================] - 7s 15ms/step - loss: 1.2750 -
accuracy: 0.5300 - val_loss: 1.2818 - val_accuracy: 0.5120
Epoch 11/20
450/450 [==============================] - 6s 14ms/step - loss: 1.2497 -
accuracy: 0.5438 - val_loss: 1.2423 - val_accuracy: 0.5460
Epoch 12/20
450/450 [==============================] - 7s 16ms/step - loss: 1.2243 -
accuracy: 0.5464 - val_loss: 1.3115 - val_accuracy: 0.5020
Epoch 13/20
450/450 [==============================] - 6s 13ms/step - loss: 1.1555 -
accuracy: 0.5647 - val_loss: 1.3146 - val_accuracy: 0.5500
Epoch 14/20
450/450 [==============================] - 7s 15ms/step - loss: 1.1490 -
accuracy: 0.5751 - val_loss: 1.1561 - val_accuracy: 0.5840
Epoch 15/20
450/450 [==============================] - 6s 14ms/step - loss: 1.1106 -
accuracy: 0.5922 - val_loss: 1.1359 - val_accuracy: 0.5620
Epoch 16/20
450/450 [==============================] - 7s 15ms/step - loss: 1.0876 -
```

```
accuracy: 0.6060 - val_loss: 1.1461 - val_accuracy: 0.5960
Epoch 17/20
450/450 [==============================] - 7s 15ms/step - loss: 1.0595 -
accuracy: 0.6067 - val_loss: 1.0617 - val_accuracy: 0.5940
Epoch 18/20
450/450 [==============================] - 6s 14ms/step - loss: 1.0293 -
accuracy: 0.6227 - val_loss: 1.3986 - val_accuracy: 0.5180
Epoch 19/20
450/450 [==============================] - 7s 16ms/step - loss: 0.9960 -
accuracy: 0.6313 - val_loss: 1.2651 - val_accuracy: 0.5340
Epoch 20/20
450/450 [==============================] - 6s 14ms/step - loss: 0.9764 -
accuracy: 0.6402 - val_loss: 1.5625 - val_accuracy: 0.4940
Epoch 1/20
450/450 [==============================] - 13s 15ms/step - loss: 2.2939 -
accuracy: 0.2484 - val_loss: 2.1683 - val_accuracy: 0.2580
Epoch 2/20
450/450 [==============================] - 7s 16ms/step - loss: 1.8471 -
accuracy: 0.3358 - val_loss: 3.1973 - val_accuracy: 0.2340
Epoch 3/20
450/450 [==============================] - 6s 14ms/step - loss: 1.7073 -
accuracy: 0.3767 - val_loss: 1.7782 - val_accuracy: 0.3340
Epoch 4/20
450/450 [==============================] - 7s 16ms/step - loss: 1.6182 -
accuracy: 0.3904 - val_loss: 1.7549 - val_accuracy: 0.3580
Epoch 5/20
450/450 [==============================] - 6s 14ms/step - loss: 1.5464 -
accuracy: 0.4302 - val_loss: 1.9235 - val_accuracy: 0.3160
Epoch 6/20
450/450 [==============================] - 7s 16ms/step - loss: 1.4843 -
accuracy: 0.4462 - val_loss: 1.6467 - val_accuracy: 0.4220
Epoch 7/20
450/450 [==============================] - 7s 15ms/step - loss: 1.4270 -
accuracy: 0.4691 - val_loss: 1.4983 - val_accuracy: 0.4460
Epoch 8/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3702 -
accuracy: 0.4913 - val_loss: 1.5015 - val_accuracy: 0.4580
Epoch 9/20
450/450 [==============================] - 7s 16ms/step - loss: 1.3195 -
accuracy: 0.5120 - val_loss: 1.4252 - val_accuracy: 0.4980
Epoch 10/20
450/450 [==============================] - 6s 14ms/step - loss: 1.2926 -
accuracy: 0.5222 - val_loss: 1.2697 - val_accuracy: 0.5440
Epoch 11/20
450/450 [==============================] - 7s 16ms/step - loss: 1.2392 -
accuracy: 0.5451 - val_loss: 1.2201 - val_accuracy: 0.5460
Epoch 12/20
450/450 [==============================] - 6s 14ms/step - loss: 1.2217 -
```

```
accuracy: 0.5478 - val_loss: 1.7812 - val_accuracy: 0.4320
Epoch 13/20
450/450 [==============================] - 7s 15ms/step - loss: 1.1835 -
accuracy: 0.5631 - val_loss: 1.2314 - val_accuracy: 0.6040
Epoch 14/20
450/450 [==============================] - 6s 14ms/step - loss: 1.1592 -
accuracy: 0.5776 - val_loss: 1.1544 - val_accuracy: 0.5900
Epoch 15/20
450/450 [==============================] - 7s 16ms/step - loss: 1.1336 -
accuracy: 0.5813 - val_loss: 1.1526 - val_accuracy: 0.5800
Epoch 16/20
450/450 [==============================] - 6s 14ms/step - loss: 1.0931 -
accuracy: 0.5938 - val_loss: 1.2468 - val_accuracy: 0.5540
Epoch 17/20
450/450 [==============================] - 7s 16ms/step - loss: 1.0721 -
accuracy: 0.6060 - val_loss: 1.3423 - val_accuracy: 0.5140
Epoch 18/20
450/450 [==============================] - 6s 14ms/step - loss: 1.0506 -
accuracy: 0.6147 - val_loss: 1.0784 - val_accuracy: 0.6120
Epoch 19/20
450/450 [==============================] - 6s 14ms/step - loss: 1.0364 -
accuracy: 0.6187 - val_loss: 1.1621 - val_accuracy: 0.5620
Epoch 20/20
450/450 [==============================] - 7s 16ms/step - loss: 1.0056 -
accuracy: 0.6353 - val_loss: 1.0974 - val_accuracy: 0.6100
Epoch 1/20
450/450 [==============================] - 12s 15ms/step - loss: 2.3543 -
accuracy: 0.2404 - val_loss: 1.7775 - val_accuracy: 0.3380
Epoch 2/20
450/450 [==============================] - 7s 15ms/step - loss: 1.9088 -
accuracy: 0.3011 - val_loss: 2.1206 - val_accuracy: 0.3220
Epoch 3/20
450/450 [==============================] - 6s 13ms/step - loss: 1.7246 -
accuracy: 0.3611 - val_loss: 2.2499 - val_accuracy: 0.3140
Epoch 4/20
450/450 [==============================] - 7s 15ms/step - loss: 1.6331 -
accuracy: 0.3867 - val_loss: 1.7057 - val_accuracy: 0.3740
Epoch 5/20
450/450 [==============================] - 6s 13ms/step - loss: 1.5617 -
accuracy: 0.4316 - val_loss: 1.3836 - val_accuracy: 0.4820
Epoch 6/20
450/450 [==============================] - 7s 15ms/step - loss: 1.5051 -
accuracy: 0.4416 - val_loss: 1.7223 - val_accuracy: 0.3540
Epoch 7/20
450/450 [==============================] - 6s 13ms/step - loss: 1.4299 -
accuracy: 0.4718 - val_loss: 1.6971 - val_accuracy: 0.4160
Epoch 8/20
450/450 [==============================] - 7s 15ms/step - loss: 1.3887 -
```

```
accuracy: 0.4827 - val_loss: 1.8132 - val_accuracy: 0.4420
Epoch 9/20
450/450 [==============================] - 6s 13ms/step - loss: 1.3512 -
accuracy: 0.4962 - val_loss: 1.3524 - val_accuracy: 0.5100
Epoch 10/20
450/450 [==============================] - 7s 15ms/step - loss: 1.2973 -
accuracy: 0.5198 - val_loss: 1.2045 - val_accuracy: 0.5720
Epoch 11/20
450/450 [==============================] - 6s 13ms/step - loss: 1.2749 -
accuracy: 0.5360 - val_loss: 1.1673 - val_accuracy: 0.5820
Epoch 12/20
450/450 [==============================] - 7s 15ms/step - loss: 1.2507 -
accuracy: 0.5391 - val_loss: 1.6164 - val_accuracy: 0.4500
Epoch 13/20
450/450 [==============================] - 6s 13ms/step - loss: 1.2225 -
accuracy: 0.5398 - val_loss: 1.3208 - val_accuracy: 0.5320
Epoch 14/20
450/450 [==============================] - 6s 13ms/step - loss: 1.1879 -
accuracy: 0.5640 - val_loss: 1.4733 - val_accuracy: 0.4800
Epoch 15/20
450/450 [==============================] - 6s 14ms/step - loss: 1.1834 -
accuracy: 0.5671 - val_loss: 1.2811 - val_accuracy: 0.5500
Epoch 16/20
450/450 [==============================] - 6s 14ms/step - loss: 1.1496 -
accuracy: 0.5816 - val_loss: 1.4910 - val_accuracy: 0.5220
Epoch 17/20
450/450 [==============================] - 6s 13ms/step - loss: 1.1355 -
accuracy: 0.5838 - val_loss: 1.2061 - val_accuracy: 0.5680
Epoch 18/20
450/450 [==============================] - 6s 13ms/step - loss: 1.1049 -
accuracy: 0.6051 - val_loss: 2.0394 - val_accuracy: 0.4100
Epoch 19/20
450/450 [==============================] - 7s 15ms/step - loss: 1.0898 -
accuracy: 0.5980 - val_loss: 1.4095 - val_accuracy: 0.5660
Epoch 20/20
450/450 [==============================] - 6s 13ms/step - loss: 1.0848 -
accuracy: 0.6013 - val_loss: 1.5566 - val_accuracy: 0.5040
Epoch 1/20
450/450 [==============================] - 11s 16ms/step - loss: 2.3601 -
accuracy: 0.2380 - val_loss: 2.0943 - val_accuracy: 0.2640
Epoch 2/20
450/450 [==============================] - 6s 13ms/step - loss: 1.8501 -
accuracy: 0.3280 - val_loss: 1.8725 - val_accuracy: 0.3560
Epoch 3/20
450/450 [==============================] - 6s 13ms/step - loss: 1.6969 -
accuracy: 0.3738 - val_loss: 1.6307 - val_accuracy: 0.3880
Epoch 4/20
450/450 [==============================] - 7s 15ms/step - loss: 1.5782 -
```

```
accuracy: 0.4171 - val_loss: 1.7486 - val_accuracy: 0.3700
Epoch 5/20
450/450 [==============================] - 6s 13ms/step - loss: 1.4969 -
accuracy: 0.4387 - val_loss: 1.7330 - val_accuracy: 0.4400
Epoch 6/20
450/450 [==============================] - 7s 15ms/step - loss: 1.4361 -
accuracy: 0.4642 - val_loss: 1.5843 - val_accuracy: 0.4240
Epoch 7/20
450/450 [==============================] - 6s 13ms/step - loss: 1.3584 -
accuracy: 0.4927 - val_loss: 1.6401 - val_accuracy: 0.4000
Epoch 8/20
450/450 [==============================] - 7s 15ms/step - loss: 1.3349 -
accuracy: 0.5042 - val_loss: 1.5856 - val_accuracy: 0.4600
Epoch 9/20
450/450 [==============================] - 6s 13ms/step - loss: 1.2697 -
accuracy: 0.5267 - val_loss: 1.3777 - val_accuracy: 0.5300
Epoch 10/20
450/450 [==============================] - 6s 13ms/step - loss: 1.2632 -
accuracy: 0.5320 - val_loss: 1.5808 - val_accuracy: 0.4660
Epoch 11/20
450/450 [==============================] - 7s 15ms/step - loss: 1.2074 -
accuracy: 0.5533 - val_loss: 1.2768 - val_accuracy: 0.5340
Epoch 12/20
450/450 [==============================] - 6s 13ms/step - loss: 1.1871 -
accuracy: 0.5562 - val_loss: 2.7392 - val_accuracy: 0.3100
Epoch 13/20
450/450 [==============================] - 7s 15ms/step - loss: 1.1634 -
accuracy: 0.5771 - val_loss: 1.1452 - val_accuracy: 0.5860
Epoch 14/20
450/450 [==============================] - 6s 13ms/step - loss: 1.1338 -
accuracy: 0.5884 - val_loss: 1.3755 - val_accuracy: 0.4960
Epoch 15/20
450/450 [==============================] - 7s 15ms/step - loss: 1.1107 -
accuracy: 0.5918 - val_loss: 1.3395 - val_accuracy: 0.5620
Epoch 16/20
450/450 [==============================] - 6s 13ms/step - loss: 1.0927 -
accuracy: 0.5991 - val_loss: 1.1603 - val_accuracy: 0.5940
Epoch 17/20
450/450 [==============================] - 6s 14ms/step - loss: 1.0623 -
accuracy: 0.6071 - val_loss: 1.6576 - val_accuracy: 0.4800
Epoch 18/20
450/450 [==============================] - 7s 15ms/step - loss: 1.0547 -
accuracy: 0.6171 - val_loss: 1.1621 - val_accuracy: 0.5960
Epoch 19/20
450/450 [==============================] - 7s 15ms/step - loss: 1.0483 -
accuracy: 0.6251 - val_loss: 2.0975 - val_accuracy: 0.4400
Epoch 20/20
450/450 [==============================] - 6s 13ms/step - loss: 1.0329 -
```

```
accuracy: 0.6180 - val_loss: 1.2791 - val_accuracy: 0.5660
Epoch 1/20
450/450 [==============================] - 13s 16ms/step - loss: 2.3083 -
accuracy: 0.2327 - val_loss: 1.8645 - val_accuracy: 0.2600
Epoch 2/20
450/450 [==============================] - 6s 14ms/step - loss: 1.9012 -
accuracy: 0.3202 - val_loss: 2.0664 - val_accuracy: 0.3040
Epoch 3/20
450/450 [==============================] - 6s 13ms/step - loss: 1.7378 -
accuracy: 0.3571 - val_loss: 2.1954 - val_accuracy: 0.2720
Epoch 4/20
450/450 [==============================] - 7s 15ms/step - loss: 1.6359 -
accuracy: 0.3947 - val_loss: 1.7279 - val_accuracy: 0.3760
Epoch 5/20
450/450 [==============================] - 6s 13ms/step - loss: 1.5506 -
accuracy: 0.4178 - val_loss: 1.7001 - val_accuracy: 0.4560
Epoch 6/20
450/450 [==============================] - 7s 15ms/step - loss: 1.5031 -
accuracy: 0.4451 - val_loss: 1.4683 - val_accuracy: 0.4460
Epoch 7/20
450/450 [==============================] - 6s 13ms/step - loss: 1.4313 -
accuracy: 0.4680 - val_loss: 2.4538 - val_accuracy: 0.3160
Epoch 8/20
450/450 [==============================] - 7s 16ms/step - loss: 1.3960 -
accuracy: 0.4860 - val_loss: 2.1260 - val_accuracy: 0.3120
Epoch 9/20
450/450 [==============================] - 6s 13ms/step - loss: 1.3404 -
accuracy: 0.5022 - val_loss: 1.2735 - val_accuracy: 0.5420
Epoch 10/20
450/450 [==============================] - 7s 15ms/step - loss: 1.2904 -
accuracy: 0.5251 - val_loss: 1.5240 - val_accuracy: 0.4600
Epoch 11/20
450/450 [==============================] - 6s 13ms/step - loss: 1.2704 -
accuracy: 0.5240 - val_loss: 1.4069 - val_accuracy: 0.5180
Epoch 12/20
450/450 [==============================] - 7s 15ms/step - loss: 1.2546 -
accuracy: 0.5364 - val_loss: 1.2891 - val_accuracy: 0.5320
Epoch 13/20
450/450 [==============================] - 6s 13ms/step - loss: 1.2035 -
accuracy: 0.5569 - val_loss: 1.1302 - val_accuracy: 0.5820
Epoch 14/20
450/450 [==============================] - 7s 15ms/step - loss: 1.1805 -
accuracy: 0.5624 - val_loss: 1.1863 - val_accuracy: 0.5980
Epoch 15/20
450/450 [==============================] - 6s 14ms/step - loss: 1.1629 -
accuracy: 0.5682 - val_loss: 1.2411 - val_accuracy: 0.5780
Epoch 16/20
450/450 [==============================] - 6s 13ms/step - loss: 1.1413 -
```

```
accuracy: 0.5811 - val_loss: 1.1660 - val_accuracy: 0.5820
Epoch 17/20
450/450 [==============================] - 7s 15ms/step - loss: 1.1285 -
accuracy: 0.5882 - val_loss: 1.1493 - val_accuracy: 0.5960
Epoch 18/20
450/450 [==============================] - 6s 13ms/step - loss: 1.1087 -
accuracy: 0.5942 - val_loss: 1.5144 - val_accuracy: 0.4960
Epoch 19/20
450/450 [==============================] - 7s 15ms/step - loss: 1.0755 -
accuracy: 0.6062 - val_loss: 1.2106 - val_accuracy: 0.5820
Epoch 20/20
450/450 [==============================] - 6s 14ms/step - loss: 1.0645 -
accuracy: 0.6087 - val_loss: 1.1192 - val_accuracy: 0.6040
Epoch 1/20
450/450 [==============================] - 10s 17ms/step - loss: 2.5665 -
accuracy: 0.2004 - val_loss: 2.7995 - val_accuracy: 0.1320
Epoch 2/20
450/450 [==============================] - 6s 13ms/step - loss: 2.1599 -
accuracy: 0.2693 - val_loss: 1.6630 - val_accuracy: 0.4000
Epoch 3/20
450/450 [==============================] - 7s 15ms/step - loss: 2.0276 -
accuracy: 0.2713 - val_loss: 1.7291 - val_accuracy: 0.3760
Epoch 4/20
450/450 [==============================] - 6s 13ms/step - loss: 1.8972 -
accuracy: 0.3136 - val_loss: 1.7352 - val_accuracy: 0.3560
Epoch 5/20
450/450 [==============================] - 7s 15ms/step - loss: 1.8038 -
accuracy: 0.3362 - val_loss: 1.6340 - val_accuracy: 0.4040
Epoch 6/20
450/450 [==============================] - 6s 14ms/step - loss: 1.7673 -
accuracy: 0.3569 - val_loss: 2.5412 - val_accuracy: 0.2460
Epoch 7/20
450/450 [==============================] - 7s 15ms/step - loss: 1.7641 -
accuracy: 0.3571 - val_loss: 1.5293 - val_accuracy: 0.4240
Epoch 8/20
450/450 [==============================] - 6s 13ms/step - loss: 1.7103 -
accuracy: 0.3671 - val_loss: 1.6540 - val_accuracy: 0.3840
Epoch 9/20
450/450 [==============================] - 7s 15ms/step - loss: 1.6678 -
accuracy: 0.3769 - val_loss: 1.4944 - val_accuracy: 0.4340
Epoch 10/20
450/450 [==============================] - 6s 14ms/step - loss: 1.6567 -
accuracy: 0.3891 - val_loss: 1.6068 - val_accuracy: 0.3820
Epoch 11/20
450/450 [==============================] - 6s 14ms/step - loss: 1.6441 -
accuracy: 0.3880 - val_loss: 1.6601 - val_accuracy: 0.3780
Epoch 12/20
450/450 [==============================] - 7s 15ms/step - loss: 1.6332 -
```

```
accuracy: 0.3947 - val_loss: 1.5207 - val_accuracy: 0.3900
Epoch 13/20
450/450 [==============================] - 6s 13ms/step - loss: 1.6144 -
accuracy: 0.3969 - val_loss: 1.4486 - val_accuracy: 0.4720
Epoch 14/20
450/450 [==============================] - 7s 16ms/step - loss: 1.5744 -
accuracy: 0.4033 - val_loss: 1.7728 - val_accuracy: 0.3720
Epoch 15/20
450/450 [==============================] - 6s 13ms/step - loss: 1.5797 -
accuracy: 0.4164 - val_loss: 1.5218 - val_accuracy: 0.4240
Epoch 16/20
450/450 [==============================] - 7s 16ms/step - loss: 1.5662 -
accuracy: 0.4129 - val_loss: 1.5139 - val_accuracy: 0.4200
Epoch 17/20
450/450 [==============================] - 6s 13ms/step - loss: 1.5360 -
accuracy: 0.4260 - val_loss: 1.4518 - val_accuracy: 0.4660
Epoch 18/20
450/450 [==============================] - 7s 16ms/step - loss: 1.5035 -
accuracy: 0.4391 - val_loss: 1.4066 - val_accuracy: 0.4580
Epoch 19/20
450/450 [==============================] - 7s 15ms/step - loss: 1.5192 -
accuracy: 0.4420 - val_loss: 1.6251 - val_accuracy: 0.4160
Epoch 20/20
450/450 [==============================] - 6s 13ms/step - loss: 1.4908 -
accuracy: 0.4498 - val_loss: 1.5508 - val_accuracy: 0.4220
Epoch 1/20
450/450 [==============================] - 9s 14ms/step - loss: 2.5868 -
accuracy: 0.2100 - val_loss: 2.1528 - val_accuracy: 0.1960
Epoch 2/20
450/450 [==============================] - 6s 13ms/step - loss: 2.2179 -
accuracy: 0.2629 - val_loss: 1.6317 - val_accuracy: 0.3780
Epoch 3/20
450/450 [==============================] - 7s 15ms/step - loss: 1.9905 -
accuracy: 0.2991 - val_loss: 1.7236 - val_accuracy: 0.3360
Epoch 4/20
450/450 [==============================] - 6s 13ms/step - loss: 1.8963 -
accuracy: 0.3136 - val_loss: 1.7423 - val_accuracy: 0.3460
Epoch 5/20
450/450 [==============================] - 7s 15ms/step - loss: 1.8472 -
accuracy: 0.3282 - val_loss: 1.5677 - val_accuracy: 0.4000
Epoch 6/20
450/450 [==============================] - 6s 13ms/step - loss: 1.7843 -
accuracy: 0.3484 - val_loss: 1.7485 - val_accuracy: 0.3920
Epoch 7/20
450/450 [==============================] - 7s 15ms/step - loss: 1.7513 -
accuracy: 0.3647 - val_loss: 1.7023 - val_accuracy: 0.3800
Epoch 8/20
450/450 [==============================] - 6s 13ms/step - loss: 1.6989 -
```

```
accuracy: 0.3642 - val_loss: 1.7353 - val_accuracy: 0.3620
Epoch 9/20
450/450 [==============================] - 7s 15ms/step - loss: 1.6759 -
accuracy: 0.3716 - val_loss: 1.7685 - val_accuracy: 0.3800
Epoch 10/20
450/450 [==============================] - 6s 13ms/step - loss: 1.6589 -
accuracy: 0.3851 - val_loss: 1.3760 - val_accuracy: 0.4920
Epoch 11/20
450/450 [==============================] - 7s 16ms/step - loss: 1.6415 -
accuracy: 0.3884 - val_loss: 1.4699 - val_accuracy: 0.4480
Epoch 12/20
450/450 [==============================] - 6s 14ms/step - loss: 1.6010 -
accuracy: 0.4104 - val_loss: 1.4315 - val_accuracy: 0.4620
Epoch 13/20
450/450 [==============================] - 6s 14ms/step - loss: 1.5951 -
accuracy: 0.4011 - val_loss: 1.4784 - val_accuracy: 0.4180
Epoch 14/20
450/450 [==============================] - 7s 14ms/step - loss: 1.5801 -
accuracy: 0.4069 - val_loss: 1.4735 - val_accuracy: 0.4640
Epoch 15/20
450/450 [==============================] - 6s 13ms/step - loss: 1.5677 -
accuracy: 0.4193 - val_loss: 1.3822 - val_accuracy: 0.4620
Epoch 16/20
450/450 [==============================] - 7s 15ms/step - loss: 1.5554 -
accuracy: 0.4267 - val_loss: 1.3892 - val_accuracy: 0.4860
Epoch 17/20
450/450 [==============================] - 7s 16ms/step - loss: 1.5514 -
accuracy: 0.4191 - val_loss: 1.3401 - val_accuracy: 0.5080
Epoch 18/20
450/450 [==============================] - 6s 13ms/step - loss: 1.5021 -
accuracy: 0.4349 - val_loss: 1.3720 - val_accuracy: 0.5160
Epoch 19/20
450/450 [==============================] - 6s 14ms/step - loss: 1.4916 -
accuracy: 0.4416 - val_loss: 1.3114 - val_accuracy: 0.5140
Epoch 20/20
450/450 [==============================] - 7s 16ms/step - loss: 1.4967 -
accuracy: 0.4387 - val_loss: 1.2931 - val_accuracy: 0.5140
Epoch 1/20
450/450 [==============================] - 10s 17ms/step - loss: 2.5633 -
accuracy: 0.2144 - val_loss: 2.0449 - val_accuracy: 0.3020
Epoch 2/20
450/450 [==============================] - 7s 15ms/step - loss: 2.1736 -
accuracy: 0.2727 - val_loss: 1.6892 - val_accuracy: 0.3460
Epoch 3/20
450/450 [==============================] - 6s 13ms/step - loss: 2.0044 -
accuracy: 0.3058 - val_loss: 2.1766 - val_accuracy: 0.2500
Epoch 4/20
450/450 [==============================] - 6s 13ms/step - loss: 1.8685 -
```

```
accuracy: 0.3278 - val_loss: 1.5869 - val_accuracy: 0.3880
Epoch 5/20
450/450 [==============================] - 7s 15ms/step - loss: 1.8238 -
accuracy: 0.3358 - val_loss: 1.7463 - val_accuracy: 0.3340
Epoch 6/20
450/450 [==============================] - 6s 13ms/step - loss: 1.7783 -
accuracy: 0.3460 - val_loss: 1.7712 - val_accuracy: 0.3720
Epoch 7/20
450/450 [==============================] - 7s 15ms/step - loss: 1.7539 -
accuracy: 0.3636 - val_loss: 1.7977 - val_accuracy: 0.3500
Epoch 8/20
450/450 [==============================] - 6s 13ms/step - loss: 1.7213 -
accuracy: 0.3693 - val_loss: 2.0569 - val_accuracy: 0.3560
Epoch 9/20
450/450 [==============================] - 7s 16ms/step - loss: 1.6760 -
accuracy: 0.3740 - val_loss: 1.4328 - val_accuracy: 0.4520
Epoch 10/20
450/450 [==============================] - 6s 13ms/step - loss: 1.6529 -
accuracy: 0.3871 - val_loss: 1.8422 - val_accuracy: 0.3800
Epoch 11/20
450/450 [==============================] - 7s 15ms/step - loss: 1.6437 -
accuracy: 0.3967 - val_loss: 1.6570 - val_accuracy: 0.4060
Epoch 12/20
450/450 [==============================] - 6s 13ms/step - loss: 1.6272 -
accuracy: 0.3929 - val_loss: 1.5762 - val_accuracy: 0.4140
Epoch 13/20
450/450 [==============================] - 7s 15ms/step - loss: 1.5855 -
accuracy: 0.4127 - val_loss: 1.5994 - val_accuracy: 0.3940
Epoch 14/20
450/450 [==============================] - 7s 15ms/step - loss: 1.5985 -
accuracy: 0.4113 - val_loss: 1.6559 - val_accuracy: 0.4180
Epoch 15/20
450/450 [==============================] - 6s 13ms/step - loss: 1.5858 -
accuracy: 0.4049 - val_loss: 1.4932 - val_accuracy: 0.4440
Epoch 16/20
450/450 [==============================] - 7s 15ms/step - loss: 1.5470 -
accuracy: 0.4116 - val_loss: 1.5915 - val_accuracy: 0.4500
Epoch 17/20
450/450 [==============================] - 6s 14ms/step - loss: 1.5313 -
accuracy: 0.4236 - val_loss: 1.5504 - val_accuracy: 0.4400
Epoch 18/20
450/450 [==============================] - 6s 13ms/step - loss: 1.4994 -
accuracy: 0.4453 - val_loss: 1.4782 - val_accuracy: 0.4360
Epoch 19/20
450/450 [==============================] - 6s 13ms/step - loss: 1.4963 -
accuracy: 0.4382 - val_loss: 1.6498 - val_accuracy: 0.4000
Epoch 20/20
450/450 [==============================] - 7s 15ms/step - loss: 1.5019 -
```

```
accuracy: 0.4404 - val_loss: 1.3240 - val_accuracy: 0.5000
Epoch 1/20
450/450 [==============================] - 13s 15ms/step - loss: 2.5475 -
accuracy: 0.2322 - val_loss: 1.9661 - val_accuracy: 0.2860
Epoch 2/20
450/450 [==============================] - 7s 16ms/step - loss: 2.0037 -
accuracy: 0.2927 - val_loss: 1.7624 - val_accuracy: 0.3040
Epoch 3/20
450/450 [==============================] - 6s 14ms/step - loss: 1.8329 -
accuracy: 0.3260 - val_loss: 1.6290 - val_accuracy: 0.3680
Epoch 4/20
450/450 [==============================] - 7s 16ms/step - loss: 1.7437 -
accuracy: 0.3436 - val_loss: 1.7639 - val_accuracy: 0.3800
Epoch 5/20
450/450 [==============================] - 6s 14ms/step - loss: 1.6700 -
accuracy: 0.3731 - val_loss: 1.6377 - val_accuracy: 0.3880
Epoch 6/20
450/450 [==============================] - 7s 15ms/step - loss: 1.6099 -
accuracy: 0.3949 - val_loss: 1.7196 - val_accuracy: 0.3780
Epoch 7/20
450/450 [==============================] - 7s 16ms/step - loss: 1.6016 -
accuracy: 0.3953 - val_loss: 1.5057 - val_accuracy: 0.4420
Epoch 8/20
450/450 [==============================] - 7s 16ms/step - loss: 1.5423 -
accuracy: 0.4182 - val_loss: 1.4811 - val_accuracy: 0.4620
Epoch 9/20
450/450 [==============================] - 6s 14ms/step - loss: 1.5071 -
accuracy: 0.4367 - val_loss: 1.5056 - val_accuracy: 0.4460
Epoch 10/20
450/450 [==============================] - 7s 16ms/step - loss: 1.4809 -
accuracy: 0.4460 - val_loss: 1.4777 - val_accuracy: 0.4440
Epoch 11/20
450/450 [==============================] - 6s 14ms/step - loss: 1.4764 -
accuracy: 0.4491 - val_loss: 1.5137 - val_accuracy: 0.4460
Epoch 12/20
450/450 [==============================] - 7s 14ms/step - loss: 1.4597 -
accuracy: 0.4536 - val_loss: 1.3566 - val_accuracy: 0.4760
Epoch 13/20
450/450 [==============================] - 7s 16ms/step - loss: 1.4153 -
accuracy: 0.4813 - val_loss: 1.5391 - val_accuracy: 0.4540
Epoch 14/20
450/450 [==============================] - 7s 15ms/step - loss: 1.3857 -
accuracy: 0.4780 - val_loss: 1.6236 - val_accuracy: 0.3980
Epoch 15/20
450/450 [==============================] - 7s 15ms/step - loss: 1.3833 -
accuracy: 0.4831 - val_loss: 1.5213 - val_accuracy: 0.4600
Epoch 16/20
450/450 [==============================] - 7s 16ms/step - loss: 1.3632 -
```

```
accuracy: 0.4909 - val_loss: 1.2884 - val_accuracy: 0.5280
Epoch 17/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3263 -
accuracy: 0.5042 - val_loss: 1.3422 - val_accuracy: 0.5140
Epoch 18/20
450/450 [==============================] - 7s 16ms/step - loss: 1.2948 -
accuracy: 0.5133 - val_loss: 1.5975 - val_accuracy: 0.4600
Epoch 19/20
450/450 [==============================] - 7s 15ms/step - loss: 1.3082 -
accuracy: 0.5164 - val_loss: 1.3008 - val_accuracy: 0.5180
Epoch 20/20
450/450 [==============================] - 6s 14ms/step - loss: 1.2771 -
accuracy: 0.5191 - val_loss: 1.0809 - val_accuracy: 0.6100
Epoch 1/20
450/450 [==============================] - 13s 17ms/step - loss: 2.5657 -
accuracy: 0.2100 - val_loss: 2.5462 - val_accuracy: 0.1740
Epoch 2/20
450/450 [==============================] - 7s 15ms/step - loss: 2.0054 -
accuracy: 0.2873 - val_loss: 2.4650 - val_accuracy: 0.2320
Epoch 3/20
450/450 [==============================] - 7s 16ms/step - loss: 1.8632 -
accuracy: 0.3136 - val_loss: 1.7847 - val_accuracy: 0.3580
Epoch 4/20
450/450 [==============================] - 6s 14ms/step - loss: 1.7553 -
accuracy: 0.3382 - val_loss: 1.5284 - val_accuracy: 0.4060
Epoch 5/20
450/450 [==============================] - 7s 15ms/step - loss: 1.7323 -
accuracy: 0.3460 - val_loss: 1.4928 - val_accuracy: 0.4180
Epoch 6/20
450/450 [==============================] - 7s 16ms/step - loss: 1.6570 -
accuracy: 0.3769 - val_loss: 1.8045 - val_accuracy: 0.3000
Epoch 7/20
450/450 [==============================] - 7s 16ms/step - loss: 1.6327 -
accuracy: 0.3898 - val_loss: 1.3850 - val_accuracy: 0.4760
Epoch 8/20
450/450 [==============================] - 7s 14ms/step - loss: 1.5809 -
accuracy: 0.4078 - val_loss: 1.7877 - val_accuracy: 0.3380
Epoch 9/20
450/450 [==============================] - 7s 14ms/step - loss: 1.5444 -
accuracy: 0.4078 - val_loss: 1.4520 - val_accuracy: 0.4520
Epoch 10/20
450/450 [==============================] - 7s 16ms/step - loss: 1.5166 -
accuracy: 0.4187 - val_loss: 1.3696 - val_accuracy: 0.5100
Epoch 11/20
450/450 [==============================] - 7s 15ms/step - loss: 1.4916 -
accuracy: 0.4422 - val_loss: 1.5882 - val_accuracy: 0.4400
Epoch 12/20
450/450 [==============================] - 7s 16ms/step - loss: 1.4805 -
```

```
accuracy: 0.4471 - val_loss: 1.3564 - val_accuracy: 0.4980
Epoch 13/20
450/450 [==============================] - 7s 16ms/step - loss: 1.4377 -
accuracy: 0.4627 - val_loss: 1.5282 - val_accuracy: 0.4320
Epoch 14/20
450/450 [==============================] - 7s 15ms/step - loss: 1.4299 -
accuracy: 0.4609 - val_loss: 1.6250 - val_accuracy: 0.4520
Epoch 15/20
450/450 [==============================] - 7s 17ms/step - loss: 1.4097 -
accuracy: 0.4822 - val_loss: 1.3405 - val_accuracy: 0.5060
Epoch 16/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3898 -
accuracy: 0.4829 - val_loss: 1.6742 - val_accuracy: 0.4020
Epoch 17/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3480 -
accuracy: 0.4962 - val_loss: 1.2651 - val_accuracy: 0.5220
Epoch 18/20
450/450 [==============================] - 7s 16ms/step - loss: 1.3302 -
accuracy: 0.5049 - val_loss: 1.4665 - val_accuracy: 0.4440
Epoch 19/20
450/450 [==============================] - 7s 16ms/step - loss: 1.3113 -
accuracy: 0.5200 - val_loss: 1.1807 - val_accuracy: 0.5600
Epoch 20/20
450/450 [==============================] - 7s 15ms/step - loss: 1.3057 -
accuracy: 0.5149 - val_loss: 1.4363 - val_accuracy: 0.5060
Epoch 1/20
450/450 [==============================] - 14s 15ms/step - loss: 2.5873 -
accuracy: 0.1973 - val_loss: 2.6974 - val_accuracy: 0.2340
Epoch 2/20
450/450 [==============================] - 7s 16ms/step - loss: 2.0393 -
accuracy: 0.2764 - val_loss: 1.9451 - val_accuracy: 0.2580
Epoch 3/20
450/450 [==============================] - 7s 16ms/step - loss: 1.8640 -
accuracy: 0.3107 - val_loss: 1.6483 - val_accuracy: 0.3540
Epoch 4/20
450/450 [==============================] - 7s 15ms/step - loss: 1.7776 -
accuracy: 0.3322 - val_loss: 1.5553 - val_accuracy: 0.3860
Epoch 5/20
450/450 [==============================] - 8s 17ms/step - loss: 1.7089 -
accuracy: 0.3567 - val_loss: 2.2110 - val_accuracy: 0.2840
Epoch 6/20
450/450 [==============================] - 7s 17ms/step - loss: 1.6678 -
accuracy: 0.3760 - val_loss: 1.6545 - val_accuracy: 0.3700
Epoch 7/20
450/450 [==============================] - 7s 16ms/step - loss: 1.6316 -
accuracy: 0.3891 - val_loss: 1.6232 - val_accuracy: 0.4020
Epoch 8/20
450/450 [==============================] - 7s 15ms/step - loss: 1.5892 -
```

```
accuracy: 0.4027 - val_loss: 1.4825 - val_accuracy: 0.4700
Epoch 9/20
450/450 [==============================] - 7s 17ms/step - loss: 1.5640 -
accuracy: 0.4118 - val_loss: 1.9248 - val_accuracy: 0.3960
Epoch 10/20
450/450 [==============================] - 7s 15ms/step - loss: 1.5277 -
accuracy: 0.4244 - val_loss: 1.5967 - val_accuracy: 0.3860
Epoch 11/20
450/450 [==============================] - 7s 15ms/step - loss: 1.4979 -
accuracy: 0.4462 - val_loss: 1.3474 - val_accuracy: 0.5100
Epoch 12/20
450/450 [==============================] - 7s 15ms/step - loss: 1.4672 -
accuracy: 0.4422 - val_loss: 1.3908 - val_accuracy: 0.4540
Epoch 13/20
450/450 [==============================] - 7s 17ms/step - loss: 1.4392 -
accuracy: 0.4698 - val_loss: 1.3484 - val_accuracy: 0.5020
Epoch 14/20
450/450 [==============================] - 7s 15ms/step - loss: 1.4286 -
accuracy: 0.4560 - val_loss: 1.3638 - val_accuracy: 0.4860
Epoch 15/20
450/450 [==============================] - 7s 15ms/step - loss: 1.4078 -
accuracy: 0.4756 - val_loss: 1.3330 - val_accuracy: 0.4920
Epoch 16/20
450/450 [==============================] - 7s 17ms/step - loss: 1.3776 -
accuracy: 0.4820 - val_loss: 1.4396 - val_accuracy: 0.4880
Epoch 17/20
450/450 [==============================] - 7s 15ms/step - loss: 1.3573 -
accuracy: 0.4913 - val_loss: 1.1705 - val_accuracy: 0.5580
Epoch 18/20
450/450 [==============================] - 8s 17ms/step - loss: 1.3345 -
accuracy: 0.4984 - val_loss: 1.2380 - val_accuracy: 0.5140
Epoch 19/20
450/450 [==============================] - 7s 16ms/step - loss: 1.3246 -
accuracy: 0.5036 - val_loss: 1.3344 - val_accuracy: 0.5100
Epoch 20/20
450/450 [==============================] - 7s 15ms/step - loss: 1.3069 -
accuracy: 0.5136 - val_loss: 1.4068 - val_accuracy: 0.4900
Epoch 1/20
450/450 [==============================] - 12s 17ms/step - loss: 2.5791 -
accuracy: 0.2207 - val_loss: 1.8342 - val_accuracy: 0.3040
Epoch 2/20
450/450 [==============================] - 7s 15ms/step - loss: 2.0722 -
accuracy: 0.2864 - val_loss: 2.2624 - val_accuracy: 0.2480
Epoch 3/20
450/450 [==============================] - 7s 16ms/step - loss: 1.8987 -
accuracy: 0.3171 - val_loss: 2.6109 - val_accuracy: 0.2660
Epoch 4/20
450/450 [==============================] - 7s 15ms/step - loss: 1.7938 -
```

```
accuracy: 0.3362 - val_loss: 1.6085 - val_accuracy: 0.3980
Epoch 5/20
450/450 [==============================] - 7s 16ms/step - loss: 1.7234 -
accuracy: 0.3649 - val_loss: 1.5861 - val_accuracy: 0.4420
Epoch 6/20
450/450 [==============================] - 7s 16ms/step - loss: 1.6852 -
accuracy: 0.3780 - val_loss: 1.5338 - val_accuracy: 0.4000
Epoch 7/20
450/450 [==============================] - 6s 14ms/step - loss: 1.6515 -
accuracy: 0.3909 - val_loss: 1.5358 - val_accuracy: 0.4180
Epoch 8/20
450/450 [==============================] - 7s 16ms/step - loss: 1.6000 -
accuracy: 0.4051 - val_loss: 1.4676 - val_accuracy: 0.4420
Epoch 9/20
450/450 [==============================] - 7s 16ms/step - loss: 1.5683 -
accuracy: 0.4142 - val_loss: 1.4970 - val_accuracy: 0.4600
Epoch 10/20
450/450 [==============================] - 6s 14ms/step - loss: 1.5530 -
accuracy: 0.4240 - val_loss: 1.5933 - val_accuracy: 0.3920
Epoch 11/20
450/450 [==============================] - 6s 14ms/step - loss: 1.5369 -
accuracy: 0.4238 - val_loss: 1.4796 - val_accuracy: 0.4700
Epoch 12/20
450/450 [==============================] - 7s 16ms/step - loss: 1.5044 -
accuracy: 0.4396 - val_loss: 1.5923 - val_accuracy: 0.4040
Epoch 13/20
450/450 [==============================] - 6s 14ms/step - loss: 1.4762 -
accuracy: 0.4464 - val_loss: 1.3921 - val_accuracy: 0.4740
Epoch 14/20
450/450 [==============================] - 7s 16ms/step - loss: 1.4877 -
accuracy: 0.4509 - val_loss: 1.2703 - val_accuracy: 0.5260
Epoch 15/20
450/450 [==============================] - 6s 14ms/step - loss: 1.4654 -
accuracy: 0.4602 - val_loss: 1.2739 - val_accuracy: 0.5120
Epoch 16/20
450/450 [==============================] - 7s 15ms/step - loss: 1.4482 -
accuracy: 0.4549 - val_loss: 1.4724 - val_accuracy: 0.4960
Epoch 17/20
450/450 [==============================] - 7s 16ms/step - loss: 1.4216 -
accuracy: 0.4776 - val_loss: 1.2853 - val_accuracy: 0.5240
Epoch 18/20
450/450 [==============================] - 6s 14ms/step - loss: 1.4066 -
accuracy: 0.4816 - val_loss: 2.5824 - val_accuracy: 0.3540
Epoch 19/20
450/450 [==============================] - 7s 16ms/step - loss: 1.3969 -
accuracy: 0.4782 - val_loss: 1.3408 - val_accuracy: 0.5160
Epoch 20/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3933 -
```

```
accuracy: 0.4902 - val_loss: 1.2774 - val_accuracy: 0.5500
Epoch 1/20
450/450 [==============================] - 12s 16ms/step - loss: 2.5922 -
accuracy: 0.2224 - val_loss: 1.9549 - val_accuracy: 0.2660
Epoch 2/20
450/450 [==============================] - 7s 16ms/step - loss: 2.0475 -
accuracy: 0.2849 - val_loss: 1.9337 - val_accuracy: 0.3220
Epoch 3/20
450/450 [==============================] - 6s 14ms/step - loss: 1.8982 -
accuracy: 0.3087 - val_loss: 1.6656 - val_accuracy: 0.3800
Epoch 4/20
450/450 [==============================] - 7s 16ms/step - loss: 1.8048 -
accuracy: 0.3402 - val_loss: 2.5178 - val_accuracy: 0.2320
Epoch 5/20
450/450 [==============================] - 6s 14ms/step - loss: 1.7365 -
accuracy: 0.3658 - val_loss: 3.3059 - val_accuracy: 0.1440
Epoch 6/20
450/450 [==============================] - 6s 14ms/step - loss: 1.6624 -
accuracy: 0.3840 - val_loss: 1.9980 - val_accuracy: 0.3060
Epoch 7/20
450/450 [==============================] - 7s 16ms/step - loss: 1.6347 -
accuracy: 0.3853 - val_loss: 2.1807 - val_accuracy: 0.2860
Epoch 8/20
450/450 [==============================] - 7s 15ms/step - loss: 1.5959 -
accuracy: 0.4122 - val_loss: 1.5730 - val_accuracy: 0.4040
Epoch 9/20
450/450 [==============================] - 7s 15ms/step - loss: 1.5646 -
accuracy: 0.4184 - val_loss: 1.4593 - val_accuracy: 0.4920
Epoch 10/20
450/450 [==============================] - 6s 14ms/step - loss: 1.5366 -
accuracy: 0.4271 - val_loss: 2.6299 - val_accuracy: 0.3360
Epoch 11/20
450/450 [==============================] - 7s 16ms/step - loss: 1.5113 -
accuracy: 0.4444 - val_loss: 2.8390 - val_accuracy: 0.2900
Epoch 12/20
450/450 [==============================] - 6s 14ms/step - loss: 1.4816 -
accuracy: 0.4518 - val_loss: 1.3703 - val_accuracy: 0.4900
Epoch 13/20
450/450 [==============================] - 7s 16ms/step - loss: 1.4534 -
accuracy: 0.4611 - val_loss: 2.0454 - val_accuracy: 0.3460
Epoch 14/20
450/450 [==============================] - 6s 14ms/step - loss: 1.4404 -
accuracy: 0.4598 - val_loss: 1.3010 - val_accuracy: 0.5260
Epoch 15/20
450/450 [==============================] - 6s 14ms/step - loss: 1.4310 -
accuracy: 0.4793 - val_loss: 1.3318 - val_accuracy: 0.5240
Epoch 16/20
450/450 [==============================] - 7s 16ms/step - loss: 1.4079 -
```

```
accuracy: 0.4702 - val_loss: 1.3448 - val_accuracy: 0.5280
Epoch 17/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3968 -
accuracy: 0.4804 - val_loss: 1.3468 - val_accuracy: 0.5020
Epoch 18/20
450/450 [==============================] - 7s 16ms/step - loss: 1.3760 -
accuracy: 0.4844 - val_loss: 1.1821 - val_accuracy: 0.5540
Epoch 19/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3759 -
accuracy: 0.4938 - val_loss: 1.5980 - val_accuracy: 0.4240
Epoch 20/20
450/450 [==============================] - 7s 16ms/step - loss: 1.3675 -
accuracy: 0.4978 - val_loss: 1.8367 - val_accuracy: 0.3760
Epoch 1/20
450/450 [==============================] - 12s 15ms/step - loss: 2.5707 -
accuracy: 0.2224 - val_loss: 4.9271 - val_accuracy: 0.1920
Epoch 2/20
450/450 [==============================] - 7s 15ms/step - loss: 2.0397 -
accuracy: 0.2838 - val_loss: 2.7546 - val_accuracy: 0.2740
Epoch 3/20
450/450 [==============================] - 6s 14ms/step - loss: 1.8596 -
accuracy: 0.3176 - val_loss: 1.7542 - val_accuracy: 0.3240
Epoch 4/20
450/450 [==============================] - 7s 16ms/step - loss: 1.7611 -
accuracy: 0.3373 - val_loss: 1.7150 - val_accuracy: 0.3300
Epoch 5/20
450/450 [==============================] - 7s 15ms/step - loss: 1.7109 -
accuracy: 0.3618 - val_loss: 1.8955 - val_accuracy: 0.3340
Epoch 6/20
450/450 [==============================] - 6s 14ms/step - loss: 1.6648 -
accuracy: 0.3858 - val_loss: 1.4522 - val_accuracy: 0.4320
Epoch 7/20
450/450 [==============================] - 6s 13ms/step - loss: 1.6135 -
accuracy: 0.3918 - val_loss: 1.5633 - val_accuracy: 0.3880
Epoch 8/20
450/450 [==============================] - 7s 16ms/step - loss: 1.5989 -
accuracy: 0.4089 - val_loss: 1.4988 - val_accuracy: 0.4460
Epoch 9/20
450/450 [==============================] - 7s 15ms/step - loss: 1.5539 -
accuracy: 0.4207 - val_loss: 1.5793 - val_accuracy: 0.4220
Epoch 10/20
450/450 [==============================] - 6s 14ms/step - loss: 1.5289 -
accuracy: 0.4300 - val_loss: 1.8832 - val_accuracy: 0.3600
Epoch 11/20
450/450 [==============================] - 7s 16ms/step - loss: 1.5053 -
accuracy: 0.4380 - val_loss: 2.0407 - val_accuracy: 0.3360
Epoch 12/20
450/450 [==============================] - 7s 16ms/step - loss: 1.4815 -
```

```
accuracy: 0.4473 - val_loss: 2.8356 - val_accuracy: 0.3020
Epoch 13/20
450/450 [==============================] - 7s 15ms/step - loss: 1.4644 -
accuracy: 0.4593 - val_loss: 1.5464 - val_accuracy: 0.4240
Epoch 14/20
450/450 [==============================] - 6s 14ms/step - loss: 1.4515 -
accuracy: 0.4553 - val_loss: 1.2489 - val_accuracy: 0.5260
Epoch 15/20
450/450 [==============================] - 7s 16ms/step - loss: 1.4290 -
accuracy: 0.4724 - val_loss: 1.3826 - val_accuracy: 0.5120
Epoch 16/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3940 -
accuracy: 0.4822 - val_loss: 1.4307 - val_accuracy: 0.4900
Epoch 17/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3936 -
accuracy: 0.4793 - val_loss: 1.3970 - val_accuracy: 0.4600
Epoch 18/20
450/450 [==============================] - 7s 16ms/step - loss: 1.3632 -
accuracy: 0.4960 - val_loss: 1.3327 - val_accuracy: 0.5140
Epoch 19/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3593 -
accuracy: 0.5033 - val_loss: 1.2468 - val_accuracy: 0.5280
Epoch 20/20
450/450 [==============================] - 6s 14ms/step - loss: 1.3635 -
accuracy: 0.5027 - val_loss: 1.3052 - val_accuracy: 0.5040
Best Parameters: {'drop_rate': 0.5, 'optimizer': 'rmsprop', 'alpha': 0.5}
Best Validation Performance: 0
```

[ ]:
```python
print(best_val_performance)
```

```
0.527999997138977
```

parameters: {‘drop_rate’: 0.5, ‘optimizer’: ‘rmsprop’, ‘alpha’: 0.5} validation accuracy:0.528

[ ]:
```python
model_3_7_3 = YourModel(name='q3_7_3',
                              feature_maps=32,
                              num_classes=data_manager.n_classes,
                              num_blocks=4,
                              drop_rate=0.5,
                              batch_norm=True,
                              is_augmentation=True,
                              use_skip=True,
                              optimizer='rmsprop',
                              num_epochs=100,
                              learning_rate=0.001)

            # Build and train the model
model_3_7_3.build_cnn()
```

```python
model_3_7_3.fit(data_manager, use_mixup=True, alpha=0.5)
```

```
Epoch 1/100
450/450 [==============================] - 10s 13ms/step - loss: 2.5998 -
accuracy: 0.2149 - val_loss: 2.2869 - val_accuracy: 0.2760
Epoch 2/100
450/450 [==============================] - 6s 14ms/step - loss: 2.0635 -
accuracy: 0.2860 - val_loss: 4.0022 - val_accuracy: 0.2400
Epoch 3/100
450/450 [==============================] - 6s 12ms/step - loss: 1.9129 -
accuracy: 0.3156 - val_loss: 2.3024 - val_accuracy: 0.3080
Epoch 4/100
450/450 [==============================] - 6s 14ms/step - loss: 1.7950 -
accuracy: 0.3300 - val_loss: 1.6124 - val_accuracy: 0.3740
Epoch 5/100
450/450 [==============================] - 6s 12ms/step - loss: 1.7564 -
accuracy: 0.3529 - val_loss: 1.6863 - val_accuracy: 0.3440
Epoch 6/100
450/450 [==============================] - 6s 13ms/step - loss: 1.6869 -
accuracy: 0.3771 - val_loss: 1.5898 - val_accuracy: 0.3840
Epoch 7/100
450/450 [==============================] - 6s 14ms/step - loss: 1.6699 -
accuracy: 0.3789 - val_loss: 1.5121 - val_accuracy: 0.4100
Epoch 8/100
450/450 [==============================] - 6s 13ms/step - loss: 1.6155 -
accuracy: 0.4049 - val_loss: 1.9935 - val_accuracy: 0.3580
Epoch 9/100
450/450 [==============================] - 6s 12ms/step - loss: 1.5722 -
accuracy: 0.4047 - val_loss: 2.1283 - val_accuracy: 0.2840
Epoch 10/100
450/450 [==============================] - 6s 14ms/step - loss: 1.5400 -
accuracy: 0.4204 - val_loss: 1.4777 - val_accuracy: 0.4460
Epoch 11/100
450/450 [==============================] - 6s 12ms/step - loss: 1.5519 -
accuracy: 0.4187 - val_loss: 1.4593 - val_accuracy: 0.4300
Epoch 12/100
450/450 [==============================] - 7s 15ms/step - loss: 1.5240 -
accuracy: 0.4273 - val_loss: 2.0205 - val_accuracy: 0.3300
Epoch 13/100
450/450 [==============================] - 6s 13ms/step - loss: 1.4983 -
accuracy: 0.4409 - val_loss: 1.7936 - val_accuracy: 0.3660
Epoch 14/100
450/450 [==============================] - 7s 15ms/step - loss: 1.4722 -
accuracy: 0.4547 - val_loss: 3.5521 - val_accuracy: 0.2660
Epoch 15/100
450/450 [==============================] - 6s 12ms/step - loss: 1.4589 -
accuracy: 0.4660 - val_loss: 1.4798 - val_accuracy: 0.4680
```

```
Epoch 16/100
450/450 [==============================] - 6s 14ms/step - loss: 1.4256 -
accuracy: 0.4616 - val_loss: 1.3579 - val_accuracy: 0.4800
Epoch 17/100
450/450 [==============================] - 6s 13ms/step - loss: 1.4090 -
accuracy: 0.4682 - val_loss: 1.2221 - val_accuracy: 0.5460
Epoch 18/100
450/450 [==============================] - 7s 15ms/step - loss: 1.4117 -
accuracy: 0.4831 - val_loss: 1.7445 - val_accuracy: 0.3980
Epoch 19/100
450/450 [==============================] - 6s 12ms/step - loss: 1.3881 -
accuracy: 0.4824 - val_loss: 1.3782 - val_accuracy: 0.4720
Epoch 20/100
450/450 [==============================] - 6s 14ms/step - loss: 1.3933 -
accuracy: 0.4800 - val_loss: 1.4160 - val_accuracy: 0.4960
Epoch 21/100
450/450 [==============================] - 7s 15ms/step - loss: 1.3732 -
accuracy: 0.4982 - val_loss: 1.3235 - val_accuracy: 0.5220
Epoch 22/100
450/450 [==============================] - 6s 12ms/step - loss: 1.3691 -
accuracy: 0.5000 - val_loss: 1.4279 - val_accuracy: 0.4780
Epoch 23/100
450/450 [==============================] - 6s 14ms/step - loss: 1.3445 -
accuracy: 0.5056 - val_loss: 1.1892 - val_accuracy: 0.5640
Epoch 24/100
450/450 [==============================] - 6s 12ms/step - loss: 1.3320 -
accuracy: 0.5089 - val_loss: 1.4513 - val_accuracy: 0.4880
Epoch 25/100
450/450 [==============================] - 7s 15ms/step - loss: 1.3317 -
accuracy: 0.5104 - val_loss: 1.3240 - val_accuracy: 0.5220
Epoch 26/100
450/450 [==============================] - 6s 12ms/step - loss: 1.3116 -
accuracy: 0.5207 - val_loss: 1.2255 - val_accuracy: 0.5340
Epoch 27/100
450/450 [==============================] - 7s 15ms/step - loss: 1.3161 -
accuracy: 0.5162 - val_loss: 1.2525 - val_accuracy: 0.5180
Epoch 28/100
450/450 [==============================] - 6s 13ms/step - loss: 1.3053 -
accuracy: 0.5180 - val_loss: 1.1635 - val_accuracy: 0.5980
Epoch 29/100
450/450 [==============================] - 7s 15ms/step - loss: 1.2903 -
accuracy: 0.5229 - val_loss: 1.1538 - val_accuracy: 0.6000
Epoch 30/100
450/450 [==============================] - 6s 12ms/step - loss: 1.2791 -
accuracy: 0.5298 - val_loss: 1.7265 - val_accuracy: 0.4480
Epoch 31/100
450/450 [==============================] - 6s 14ms/step - loss: 1.2914 -
accuracy: 0.5269 - val_loss: 1.2148 - val_accuracy: 0.5680
```

```
Epoch 32/100
450/450 [==============================] - 6s 13ms/step - loss: 1.2689 -
accuracy: 0.5300 - val_loss: 1.1514 - val_accuracy: 0.5920
Epoch 33/100
450/450 [==============================] - 6s 14ms/step - loss: 1.2539 -
accuracy: 0.5376 - val_loss: 1.2365 - val_accuracy: 0.5460
Epoch 34/100
450/450 [==============================] - 6s 12ms/step - loss: 1.2408 -
accuracy: 0.5418 - val_loss: 1.4547 - val_accuracy: 0.5100
Epoch 35/100
450/450 [==============================] - 7s 14ms/step - loss: 1.2354 -
accuracy: 0.5482 - val_loss: 1.2517 - val_accuracy: 0.5640
Epoch 36/100
450/450 [==============================] - 6s 12ms/step - loss: 1.2478 -
accuracy: 0.5431 - val_loss: 1.1473 - val_accuracy: 0.5820
Epoch 37/100
450/450 [==============================] - 7s 14ms/step - loss: 1.2325 -
accuracy: 0.5604 - val_loss: 1.2154 - val_accuracy: 0.5660
Epoch 38/100
450/450 [==============================] - 5s 12ms/step - loss: 1.2225 -
accuracy: 0.5558 - val_loss: 1.2052 - val_accuracy: 0.5960
Epoch 39/100
450/450 [==============================] - 6s 14ms/step - loss: 1.2157 -
accuracy: 0.5533 - val_loss: 1.1928 - val_accuracy: 0.5640
Epoch 40/100
450/450 [==============================] - 6s 13ms/step - loss: 1.2312 -
accuracy: 0.5500 - val_loss: 1.1695 - val_accuracy: 0.5980
Epoch 41/100
450/450 [==============================] - 7s 15ms/step - loss: 1.1942 -
accuracy: 0.5631 - val_loss: 1.1628 - val_accuracy: 0.5740
Epoch 42/100
450/450 [==============================] - 6s 12ms/step - loss: 1.1753 -
accuracy: 0.5796 - val_loss: 1.2074 - val_accuracy: 0.6160
Epoch 43/100
450/450 [==============================] - 6s 14ms/step - loss: 1.1893 -
accuracy: 0.5649 - val_loss: 1.3253 - val_accuracy: 0.5360
Epoch 44/100
450/450 [==============================] - 6s 12ms/step - loss: 1.2337 -
accuracy: 0.5627 - val_loss: 1.2316 - val_accuracy: 0.5600
Epoch 45/100
450/450 [==============================] - 7s 15ms/step - loss: 1.1773 -
accuracy: 0.5716 - val_loss: 1.1840 - val_accuracy: 0.5620
Epoch 46/100
450/450 [==============================] - 6s 12ms/step - loss: 1.1712 -
accuracy: 0.5744 - val_loss: 1.1397 - val_accuracy: 0.6060
Epoch 47/100
450/450 [==============================] - 7s 15ms/step - loss: 1.1687 -
accuracy: 0.5778 - val_loss: 1.1424 - val_accuracy: 0.6000
```

```
Epoch 48/100
450/450 [==============================] - 6s 13ms/step - loss: 1.1828 -
accuracy: 0.5693 - val_loss: 1.1638 - val_accuracy: 0.5740
Epoch 49/100
450/450 [==============================] - 7s 14ms/step - loss: 1.1581 -
accuracy: 0.5771 - val_loss: 1.3094 - val_accuracy: 0.5580
Epoch 50/100
450/450 [==============================] - 6s 12ms/step - loss: 1.1559 -
accuracy: 0.5796 - val_loss: 1.2321 - val_accuracy: 0.5600
Epoch 51/100
450/450 [==============================] - 7s 15ms/step - loss: 1.1859 -
accuracy: 0.5742 - val_loss: 1.2658 - val_accuracy: 0.5740
Epoch 52/100
450/450 [==============================] - 6s 13ms/step - loss: 1.1670 -
accuracy: 0.5853 - val_loss: 1.4662 - val_accuracy: 0.5040
Epoch 53/100
450/450 [==============================] - 6s 14ms/step - loss: 1.1625 -
accuracy: 0.5833 - val_loss: 1.3327 - val_accuracy: 0.5740
Epoch 54/100
450/450 [==============================] - 6s 12ms/step - loss: 1.1430 -
accuracy: 0.5791 - val_loss: 1.2842 - val_accuracy: 0.5820
Epoch 55/100
450/450 [==============================] - 6s 14ms/step - loss: 1.1210 -
accuracy: 0.5909 - val_loss: 1.6690 - val_accuracy: 0.5140
Epoch 56/100
450/450 [==============================] - 6s 12ms/step - loss: 1.1395 -
accuracy: 0.5891 - val_loss: 1.2477 - val_accuracy: 0.5900
Epoch 57/100
450/450 [==============================] - 7s 15ms/step - loss: 1.1416 -
accuracy: 0.5884 - val_loss: 1.0695 - val_accuracy: 0.6340
Epoch 58/100
450/450 [==============================] - 6s 13ms/step - loss: 1.1429 -
accuracy: 0.5842 - val_loss: 1.2281 - val_accuracy: 0.5820
Epoch 59/100
450/450 [==============================] - 6s 14ms/step - loss: 1.1460 -
accuracy: 0.5898 - val_loss: 1.0874 - val_accuracy: 0.6200
Epoch 60/100
450/450 [==============================] - 7s 15ms/step - loss: 1.1250 -
accuracy: 0.5911 - val_loss: 1.1943 - val_accuracy: 0.5940
Epoch 61/100
450/450 [==============================] - 6s 12ms/step - loss: 1.1216 -
accuracy: 0.5862 - val_loss: 1.2558 - val_accuracy: 0.5500
Epoch 62/100
450/450 [==============================] - 7s 15ms/step - loss: 1.1145 -
accuracy: 0.5944 - val_loss: 1.1502 - val_accuracy: 0.5840
Epoch 63/100
450/450 [==============================] - 6s 12ms/step - loss: 1.1128 -
accuracy: 0.6020 - val_loss: 1.3086 - val_accuracy: 0.5960
```

```
Epoch 64/100
450/450 [==============================] - 7s 14ms/step - loss: 1.1131 -
accuracy: 0.5976 - val_loss: 1.2359 - val_accuracy: 0.5860
Epoch 65/100
450/450 [==============================] - 6s 12ms/step - loss: 1.1120 -
accuracy: 0.5993 - val_loss: 1.1795 - val_accuracy: 0.5780
Epoch 66/100
450/450 [==============================] - 6s 13ms/step - loss: 1.0987 -
accuracy: 0.6058 - val_loss: 1.1759 - val_accuracy: 0.6120
Epoch 67/100
450/450 [==============================] - 6s 13ms/step - loss: 1.1085 -
accuracy: 0.6016 - val_loss: 1.0956 - val_accuracy: 0.6280
Epoch 68/100
450/450 [==============================] - 6s 13ms/step - loss: 1.1028 -
accuracy: 0.5987 - val_loss: 1.1208 - val_accuracy: 0.6080
Epoch 69/100
450/450 [==============================] - 6s 13ms/step - loss: 1.1054 -
accuracy: 0.5993 - val_loss: 1.1493 - val_accuracy: 0.6220
Epoch 70/100
450/450 [==============================] - 6s 13ms/step - loss: 1.0805 -
accuracy: 0.6076 - val_loss: 1.3525 - val_accuracy: 0.5880
Epoch 71/100
450/450 [==============================] - 6s 13ms/step - loss: 1.1125 -
accuracy: 0.5902 - val_loss: 1.2326 - val_accuracy: 0.6320
Epoch 72/100
450/450 [==============================] - 6s 13ms/step - loss: 1.1127 -
accuracy: 0.5913 - val_loss: 1.0367 - val_accuracy: 0.6380
Epoch 73/100
450/450 [==============================] - 7s 15ms/step - loss: 1.0679 -
accuracy: 0.6162 - val_loss: 1.2289 - val_accuracy: 0.6060
Epoch 74/100
450/450 [==============================] - 6s 12ms/step - loss: 1.0815 -
accuracy: 0.6127 - val_loss: 1.1822 - val_accuracy: 0.5960
Epoch 75/100
450/450 [==============================] - 7s 14ms/step - loss: 1.0811 -
accuracy: 0.6062 - val_loss: 1.0663 - val_accuracy: 0.6120
Epoch 76/100
450/450 [==============================] - 7s 15ms/step - loss: 1.0974 -
accuracy: 0.6027 - val_loss: 1.0462 - val_accuracy: 0.6380
Epoch 77/100
450/450 [==============================] - 6s 12ms/step - loss: 1.0553 -
accuracy: 0.6149 - val_loss: 1.2851 - val_accuracy: 0.5880
Epoch 78/100
450/450 [==============================] - 6s 14ms/step - loss: 1.0917 -
accuracy: 0.6016 - val_loss: 1.0616 - val_accuracy: 0.6220
Epoch 79/100
450/450 [==============================] - 6s 12ms/step - loss: 1.0823 -
accuracy: 0.6118 - val_loss: 1.1079 - val_accuracy: 0.6180
```

```
Epoch 80/100
450/450 [==============================] - 7s 15ms/step - loss: 1.0740 -
accuracy: 0.6120 - val_loss: 1.2189 - val_accuracy: 0.5940
Epoch 81/100
450/450 [==============================] - 6s 13ms/step - loss: 1.0808 -
accuracy: 0.6113 - val_loss: 1.1286 - val_accuracy: 0.6260
Epoch 82/100
450/450 [==============================] - 7s 15ms/step - loss: 1.0804 -
accuracy: 0.6113 - val_loss: 1.0766 - val_accuracy: 0.6340
Epoch 83/100
450/450 [==============================] - 6s 13ms/step - loss: 1.0702 -
accuracy: 0.6171 - val_loss: 1.4214 - val_accuracy: 0.5340
Epoch 84/100
450/450 [==============================] - 7s 15ms/step - loss: 1.0908 -
accuracy: 0.6096 - val_loss: 1.4015 - val_accuracy: 0.5840
Epoch 85/100
450/450 [==============================] - 6s 12ms/step - loss: 1.0674 -
accuracy: 0.6156 - val_loss: 1.1400 - val_accuracy: 0.6180
Epoch 86/100
450/450 [==============================] - 6s 14ms/step - loss: 1.0827 -
accuracy: 0.6076 - val_loss: 1.0858 - val_accuracy: 0.6160
Epoch 87/100
450/450 [==============================] - 6s 13ms/step - loss: 1.0504 -
accuracy: 0.6213 - val_loss: 1.1134 - val_accuracy: 0.6120
Epoch 88/100
450/450 [==============================] - 6s 14ms/step - loss: 1.0759 -
accuracy: 0.6116 - val_loss: 1.1505 - val_accuracy: 0.6260
Epoch 89/100
450/450 [==============================] - 6s 12ms/step - loss: 1.0474 -
accuracy: 0.6278 - val_loss: 1.0894 - val_accuracy: 0.6600
Epoch 90/100
450/450 [==============================] - 6s 14ms/step - loss: 1.0892 -
accuracy: 0.6160 - val_loss: 1.1571 - val_accuracy: 0.6100
Epoch 91/100
450/450 [==============================] - 6s 13ms/step - loss: 1.0517 -
accuracy: 0.6313 - val_loss: 1.0817 - val_accuracy: 0.6440
Epoch 92/100
450/450 [==============================] - 6s 14ms/step - loss: 1.0614 -
accuracy: 0.6156 - val_loss: 1.4039 - val_accuracy: 0.6140
Epoch 93/100
450/450 [==============================] - 6s 12ms/step - loss: 1.0534 -
accuracy: 0.6307 - val_loss: 1.2693 - val_accuracy: 0.5960
Epoch 94/100
450/450 [==============================] - 7s 15ms/step - loss: 1.0379 -
accuracy: 0.6282 - val_loss: 1.1655 - val_accuracy: 0.6300
Epoch 95/100
450/450 [==============================] - 6s 13ms/step - loss: 1.0436 -
accuracy: 0.6209 - val_loss: 1.2218 - val_accuracy: 0.6020
```

```
Epoch 96/100
450/450 [==============================] - 6s 14ms/step - loss: 1.0629 -
accuracy: 0.6291 - val_loss: 1.1238 - val_accuracy: 0.6240
Epoch 97/100
450/450 [==============================] - 6s 12ms/step - loss: 1.0338 -
accuracy: 0.6287 - val_loss: 1.2165 - val_accuracy: 0.6160
Epoch 98/100
450/450 [==============================] - 6s 14ms/step - loss: 1.0401 -
accuracy: 0.6320 - val_loss: 1.2140 - val_accuracy: 0.6200
Epoch 99/100
450/450 [==============================] - 6s 13ms/step - loss: 1.0427 -
accuracy: 0.6313 - val_loss: 1.2121 - val_accuracy: 0.5840
Epoch 100/100
450/450 [==============================] - 7s 14ms/step - loss: 1.0303 -
accuracy: 0.6429 - val_loss: 1.2009 - val_accuracy: 0.6360
```

The best model so far parameter is: num_block=4, dropout=0.5, alpha =0.5,num_epoch=100,optimizer='rmsprop',learning_rate=0.001,use_skip=True,is_augmentation=True,mixup= Validation accuracy=0.636

```
[ ]: model_3_7_3.model.save("model_3_7_3.h5")#save the model, and this is the best␣
      ↪model found so far
```

```
[ ]: from tensorflow.keras.models import load_model

     loaded_model = load_model("model_3_7_3.h5")
```

```
[ ]: num_samples = 25
     sample_dataset = data_manager.ds_test.take(num_samples)
     model_3_7_3.predict(sample_dataset.batch(num_samples), data_manager.ds_info)#␣
      ↪test the model on test dataset
```

```
1/1 [==============================] - 1s 641ms/step
Sample 1: Predicted label - car
Sample 2: Predicted label - cat
Sample 3: Predicted label - cat
Sample 4: Predicted label - bird
Sample 5: Predicted label - monkey
Sample 6: Predicted label - cat
Sample 7: Predicted label - cat
Sample 8: Predicted label - dog
Sample 9: Predicted label - horse
Sample 10: Predicted label - deer
Sample 11: Predicted label - deer
Sample 12: Predicted label - cat
Sample 13: Predicted label - dog
Sample 14: Predicted label - ship
Sample 15: Predicted label - deer
```

```
Sample 16: Predicted label - deer
Sample 17: Predicted label - dog
Sample 18: Predicted label - airplane
Sample 19: Predicted label - airplane
Sample 20: Predicted label - monkey
Sample 21: Predicted label - horse
Sample 22: Predicted label - dog
Sample 23: Predicted label - ship
Sample 24: Predicted label - airplane
Sample 25: Predicted label - airplane
```

[ ]:
```
num_samples = 20
sample_dataset = data_manager.ds_test.take(num_samples)
model_3_7_3.plot_predictions(sample_dataset, data_manager.ds_info, num_samples=num_samples, grid_shape=(4, 5))#visualize the model on test dataset
```

```
1/1 [==============================] - 0s 78ms/step
```

In conclusion: the best model found so far is: number of block:4 learning rate:0.001 use skip_connection use data_augmentation dropout=0.5 alpha for mixup=0.5 optimizer:rmsprop number of epoches:100

And best model found has been saved to "model_3_7_3.h5"

For questions **3.8 and 3.9**, you can reuse code in lectures or labs. You should not use third-party libraries such as ClevenHans.

0.1.8 Question 3.8: Attack your model

Attack your best obtained model with PGD, MIM, and FGSM attacks with $\epsilon = 0.0313, k = 20, \eta = 0.002$ on the testing set. Write the code for the attacks and report the robust accuracies. Also choose a random set of 20 clean images in the testing set and visualize the original and attacked images.

[5 points]

```
[ ]: # YOU ARE REQUIRED TO INSERT YOUR CODES IN THIS CELL
     # You can add more cells if necessary
```

0.1.9 Question 3.9: Train a robust model

Train a robust model using adversarial training with PGD $\epsilon = 0.0313, k = 10, \eta = 0.002$. Write the code for the adversarial training and report the robust accuracies. After finishing the training, you need to store your best robust model in the folder `./models` and load the model to evaluate the robust accuracies for PGD, MIM, and FGSM attacks with $\epsilon = 0.0313, k = 20, \eta = 0.002$ on the testing set.

[5 points]

```
[ ]: # YOU ARE REQUIRED TO INSERT YOUR CODES IN THIS CELL
     # You can add more cells if necessary
```

The following is an exploring question with bonus points. It is great if you try to do this question, but it is **totally optional**. In this question, we will investigate a recent SOTA technique to improve the generalization ability of deep nets named *Sharpness-Aware Minimization (SAM)* (link to the main paper). Furthermore, SAM is simple and efficient technique, but roughly doubles the training time due to its required computation. If you have an idea to improve SAM, it would be a great paper to top-tier venues in machine learning and computer vision. Highly recommend to give it a try.

0.1.10 Question 3.10 (bonus question)

Read the SAM paper (link to the main paper). Try to apply this technique to the best obtained model and report the results. For the purpose of implementing SAM, we can flexibly add more cells and extensions to the `model.py` file.

[5 points]

[ ]:
```
# YOU ARE REQUIRED TO INSERT YOUR CODES IN THIS CELL
# You can add more cells if necessary
```

**

END OF ASSIGNMENT

GOOD LUCK WITH YOUR ASSIGNMENT 1!

**