```
from keras.models import Sequential
from keras.layers import Dense, Conv2D, MaxPooling2D, Flatten


model = Sequential()



model.add(Conv2D(64, (3, 3), activation='relu' , input_shape = (64,64,3)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(4, activation='softmax'))
```

```
/usr/local/lib/python3.11/dist-packages/keras/src/layers/convolutional/base_conv.py:107: UserWarning: Do not pass an `input_shape`/`inpu
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
```

```
model.summary()
```

```
Model: "sequential"
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┓
┃ Layer (type)                    ┃ Output Shape           ┃       Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━┩
│ conv2d (Conv2D)                 │ (None, 62, 62, 64)     │         1,792 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ max_pooling2d (MaxPooling2D)    │ (None, 31, 31, 64)     │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ flatten (Flatten)               │ (None, 61504)          │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense (Dense)                   │ (None, 128)            │     7,872,640 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense_1 (Dense)                 │ (None, 4)              │           516 │
└─────────────────────────────────┴────────────────────────┴───────────────┘
 Total params: 7,874,948 (30.04 MB)
 Trainable params: 7,874,948 (30.04 MB)
 Non-trainable params: 0 (0.00 B)
```

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   rotation_range = 0.2,
                                   width_shift_range = 0.2,
                                   height_shift_range = 0.2,
                                   fill_mode = 'nearest',
                                   vertical_flip = True,
                                   horizontal_flip = True)
test_datagen = ImageDataGenerator(rescale = 1./255)


train_path = '/content/drive/MyDrive/Dog - panda/test'
test_path =  '/content/drive/MyDrive/Dog - panda/train'
train_generator = train_datagen.flow_from_directory(train_path,
                                                    target_size = (64, 64),
                                                    batch_size = 8,
                                                    class_mode = 'categorical')
test_generator = test_datagen.flow_from_directory(test_path,
                                                    target_size = (64, 64),
                                                    batch_size = 8,
                                                    class_mode = 'categorical')
```

```
Found 14 images belonging to 4 classes.
Found 37 images belonging to 4 classes.
```

```
train_generator.class_indices
```

```
{'dog test': 0, 'dolphin test': 1, 'panda test': 2, 'penguin test': 3}
```

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

```
model.fit(train_generator, epochs=100, validation_data=test_generator)
```

```
/usr/local/lib/python3.11/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121: UserWarning: Your `PyDataset` clas
  self._warn_if_super_not_called()
Epoch 1/100
```

```
2/2 ━━━━━━━━━━━━━━━━━━━━ 18s 16s/step - accuracy: 0.0893 - loss: 7.9256 - val_accuracy: 0.3514 - val_loss: 4.7447
Epoch 2/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 4s 382ms/step - accuracy: 0.1984 - loss: 3.9148 - val_accuracy: 0.1622 - val_loss: 1.9633
Epoch 3/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 410ms/step - accuracy: 0.3155 - loss: 1.8811 - val_accuracy: 0.4595 - val_loss: 2.4832
Epoch 4/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 430ms/step - accuracy: 0.4048 - loss: 2.0782 - val_accuracy: 0.4595 - val_loss: 1.8349
Epoch 5/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 514ms/step - accuracy: 0.3155 - loss: 1.3690 - val_accuracy: 0.4595 - val_loss: 1.6179
Epoch 6/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 439ms/step - accuracy: 0.3631 - loss: 1.3631 - val_accuracy: 0.1622 - val_loss: 1.3814
Epoch 7/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 439ms/step - accuracy: 0.3571 - loss: 1.2481 - val_accuracy: 0.4054 - val_loss: 1.2748
Epoch 8/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 401ms/step - accuracy: 0.3571 - loss: 1.2066 - val_accuracy: 0.3784 - val_loss: 1.3008
Epoch 9/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 441ms/step - accuracy: 0.6310 - loss: 1.1406 - val_accuracy: 0.1622 - val_loss: 1.4653
Epoch 10/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 447ms/step - accuracy: 0.4524 - loss: 1.1345 - val_accuracy: 0.1622 - val_loss: 1.5241
Epoch 11/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 451ms/step - accuracy: 0.5952 - loss: 1.1316 - val_accuracy: 0.2703 - val_loss: 1.4626
Epoch 12/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 534ms/step - accuracy: 0.8214 - loss: 0.9316 - val_accuracy: 0.3243 - val_loss: 1.3627
Epoch 13/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 516ms/step - accuracy: 0.8016 - loss: 0.9324 - val_accuracy: 0.3243 - val_loss: 1.2883
Epoch 14/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 531ms/step - accuracy: 0.5476 - loss: 1.0123 - val_accuracy: 0.2162 - val_loss: 1.3085
Epoch 15/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 2s 832ms/step - accuracy: 0.7460 - loss: 0.9607 - val_accuracy: 0.2973 - val_loss: 1.3713
Epoch 16/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 380ms/step - accuracy: 0.5397 - loss: 0.8484 - val_accuracy: 0.3243 - val_loss: 1.4916
Epoch 17/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 407ms/step - accuracy: 0.6508 - loss: 0.8296 - val_accuracy: 0.3243 - val_loss: 1.4036
Epoch 18/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 493ms/step - accuracy: 0.8155 - loss: 0.7782 - val_accuracy: 0.3514 - val_loss: 1.4839
Epoch 19/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 350ms/step - accuracy: 0.8492 - loss: 0.6319 - val_accuracy: 0.3243 - val_loss: 1.6334
Epoch 20/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 414ms/step - accuracy: 0.7679 - loss: 0.6940 - val_accuracy: 0.3784 - val_loss: 1.5593
Epoch 21/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 558ms/step - accuracy: 0.7262 - loss: 0.6800 - val_accuracy: 0.2973 - val_loss: 1.5232
Epoch 22/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 454ms/step - accuracy: 0.8492 - loss: 0.5812 - val_accuracy: 0.2973 - val_loss: 1.7054
Epoch 23/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 482ms/step - accuracy: 0.8631 - loss: 0.4932 - val_accuracy: 0.3514 - val_loss: 2.0188
Epoch 24/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 422ms/step - accuracy: 1.0000 - loss: 0.5630 - val_accuracy: 0.3514 - val_loss: 2.0856
Epoch 25/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 414ms/step - accuracy: 0.5952 - loss: 0.6028 - val_accuracy: 0.3514 - val_loss: 2.1105
Epoch 26/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 491ms/step - accuracy: 0.8155 - loss: 0.4448 - val_accuracy: 0.3784 - val_loss: 2.2078
Epoch 27/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 467ms/step - accuracy: 1.0000 - loss: 0.4182 - val_accuracy: 0.2973 - val_loss: 1.8403
Epoch 28/100
```

```
model.save('dog-panda-dolphin-penguin-classifier.h5')
```

```
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is consi
```

Start coding or generate with AI.