Set a base_model.

```
from tensorflow.keras import datasets, layers, models, applications, optimizers

base_model = applications.DenseNet121(input_shape = (32,32,3),
                                       include_top = False,
                                       weights = 'imagenet')
```

Freeze parts of the base_model and layers specific for the new model, then train the new model.

```
(x_train, y_train), (x_test, y_test) = datasets.cifar10.load_data()

base_model.trainable = False # freeze the weights on all layers

for layer in base_model.layers[-60:]:
  layer.trainable = True

new_model = models.Sequential([
    base_model,
    layers.Flatten(),
    layers.Dense(1024, activation='relu'),
    layers.Dropout(0.1),
    layers.Dense(512, activation='relu'),
    layers.Dropout(0.2),
    layers.Dense(512, activation='relu'),
    layers.Dense(10, activation='softmax')
])

adam_low_rate = optimizers.Adam(learning_rate=0.001)

new_model.compile(optimizer = adam_low_rate,
                  loss='sparse_categorical_crossentropy',
                  metrics=['accuracy'])

history = new_model.fit(x_train, y_train, epochs=5, validation_data=(x_test, y_test), batch_size=128)
test_loss, test_acc = new_model.evaluate(x_train, y_train)
```

The result of training a new model.

```
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170500096/170498071 [==============================] - 3s 0us/step
170508288/170498071 [==============================] - 3s 0us/step
Epoch 1/5
391/391 [==============================] - 182s 448ms/step - loss: 1.5284 - accuracy: 0.4568 - val_loss: 1.3509 - val_accuracy: 0.5156
Epoch 2/5
391/391 [==============================] - 177s 452ms/step - loss: 1.3170 - accuracy: 0.5332 - val_loss: 1.3205 - val_accuracy: 0.5333
Epoch 3/5
391/391 [==============================] - 175s 449ms/step - loss: 1.2209 - accuracy: 0.5675 - val_loss: 1.2710 - val_accuracy: 0.5498
Epoch 4/5
391/391 [==============================] - 178s 455ms/step - loss: 1.1451 - accuracy: 0.5923 - val_loss: 1.2389 - val_accuracy: 0.5632
Epoch 5/5
391/391 [==============================] - 178s 455ms/step - loss: 1.0733 - accuracy: 0.6167 - val_loss: 1.2466 - val_accuracy: 0.5660
1563/1563 [==============================] - 187s 120ms/step - loss: 0.8906 - accuracy: 0.6815
```

Replace base model with VGG16 and train the new model.

```
base_model = applications.VGG16(input_shape = (32,32,3),
                                        include_top = False,
                                        weights = 'imagenet')

base_model.trainable = False # freeze the weights on all layers

for layer in base_model.layers[-60:]:
  layer.trainable = True

new_model = models.Sequential([
    base_model,
    layers.Flatten(),
    layers.Dense(512, activation='relu'),
    layers.Dropout(0.4),
    layers.Dense(512, activation='relu'),
    layers.Dense(10, activation='softmax')
])

adam_low_rate = optimizers.Adam(learning_rate=0.001)

new_model.compile(optimizer = adam_low_rate,
                  loss='sparse_categorical_crossentropy',
                  metrics=['accuracy'])

history = new_model.fit(x_train, y_train, epochs=5, validation_data=(x_test, y_test), batch_size=128)
test_loss, test_acc = new_model.evaluate(x_train, y_train)
```

The accuracy has improved.

```
Epoch 1/5
391/391 [==============================] - 551s 1s/step - loss: 2.4203 - accuracy: 0.4309 - val_loss: 1.3593 - val_accuracy: 0.5281
Epoch 2/5
391/391 [==============================] - 550s 1s/step - loss: 1.3492 - accuracy: 0.5364 - val_loss: 1.2424 - val_accuracy: 0.5780
Epoch 3/5
391/391 [==============================] - 549s 1s/step - loss: 1.2105 - accuracy: 0.5785 - val_loss: 1.1764 - val_accuracy: 0.5988
Epoch 4/5
391/391 [==============================] - 550s 1s/step - loss: 1.1274 - accuracy: 0.6094 - val_loss: 1.1525 - val_accuracy: 0.6092
Epoch 5/5
391/391 [==============================] - 552s 1s/step - loss: 1.0636 - accuracy: 0.6275 - val_loss: 1.1362 - val_accuracy: 0.6145
1563/1563 [==============================] - 483s 309ms/step - loss: 0.8980 - accuracy: 0.7036
```

Change the learning rate and epochs.

```
base_model = applications.VGG16(input_shape = (32,32,3),
                                        include_top = False,
                                        weights = 'imagenet')

base_model.trainable = False # freeze the weights on all layers

for layer in base_model.layers[-60:]:
  layer.trainable = True

new_model = models.Sequential([
    base_model,
    layers.Flatten(),
    layers.Dense(512, activation='relu'),
    layers.Dropout(0.4),
    layers.Dense(512, activation='relu'),
    layers.Dense(10, activation='softmax')
])

adam_low_rate = optimizers.Adam(learning_rate=0.002)

new_model.compile(optimizer = adam_low_rate,
                  loss='sparse_categorical_crossentropy',
                  metrics=['accuracy'])

history = new_model.fit(x_train, y_train, epochs=8, validation_data=(x_test, y_test), batch_size=128)
test_loss, test_acc = new_model.evaluate(x_train, y_train)
```

```
Epoch 1/8
391/391 [==============================] - 551s 1s/step - loss: 2.1120 - accuracy: 0.4549 - val_loss: 1.3199 - val_accuracy: 0.5467
Epoch 2/8
391/391 [==============================] - 552s 1s/step - loss: 1.2945 - accuracy: 0.5534 - val_loss: 1.2121 - val_accuracy: 0.5879
Epoch 3/8
391/391 [==============================] - 553s 1s/step - loss: 1.2083 - accuracy: 0.5840 - val_loss: 1.1845 - val_accuracy: 0.6075
Epoch 4/8
391/391 [==============================] - 552s 1s/step - loss: 1.1459 - accuracy: 0.6066 - val_loss: 1.1535 - val_accuracy: 0.6135
Epoch 5/8
391/391 [==============================] - 553s 1s/step - loss: 1.1132 - accuracy: 0.6163 - val_loss: 1.1572 - val_accuracy: 0.6071
Epoch 6/8
391/391 [==============================] - 552s 1s/step - loss: 1.0868 - accuracy: 0.6227 - val_loss: 1.1344 - val_accuracy: 0.6148
Epoch 7/8
391/391 [==============================] - 552s 1s/step - loss: 1.0605 - accuracy: 0.6320 - val_loss: 1.1298 - val_accuracy: 0.6220
Epoch 8/8
391/391 [==============================] - 553s 1s/step - loss: 1.0374 - accuracy: 0.6443 - val_loss: 1.1247 - val_accuracy: 0.6191
1563/1563 [==============================] - 485s 310ms/step - loss: 0.8888 - accuracy: 0.7043
```