Deep Learning (TensorFlow, Keras) with ResNet50: Image Binary Classifier (Part 3)

In this project, a model is trained to perform binary classifiaction for cats and dogs pictures. The pretrained model ResNet50 is used. This document is the third part of the whole training process.

Iteration 3: Model retraining with data augmentation, fine-tuning (last 20 layers) and learning_rate = 1e-5

```
# (height, width, channels)
input_shape = (224, 224, 3)
batch_size = 8
learning_rate = 1e-5
neurons = 128
path_dataset = '../dataset_cat_dogs'
folder_cat = 'Cat'
folder_dog = 'Dog'
folder_models = '../models'

import pandas as pd
import matplotlib.pyplot as plt
import os
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import ResNet50
from tensorflow.keras.layers import GlobalAveragePooling2D, Dense
from tensorflow.keras.models import Sequential
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import EarlyStopping,
ReduceLROnPlateau, ModelCheckpoint
from tensorflow.keras.models import load_model
```

```
2025-09-29 15:45:17.278411: I tensorflow/tsl/cuda/cudart_stub.cc:28]
Could not find cuda drivers on your machine, GPU will not be used.
2025-09-29 15:45:17.756154: I tensorflow/tsl/cuda/cudart_stub.cc:28]
Could not find cuda drivers on your machine, GPU will not be used.
2025-09-29 15:45:17.768662: I
tensorflow/core/platform/cpu_feature_guard.cc:182] This TensorFlow
binary is optimized to use available CPU instructions in performance-
critical operations.
To enable the following instructions: AVX2 FMA, in other operations,
rebuild TensorFlow with the appropriate compiler flags.
2025-09-29 15:45:25.353670: W
tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning:
Could not find TensorRT
```

Data augmentation

```
def load_data(path, input_shape=input_shape, batch_size=batch_size,
seed=123, validation_split=0.2):
    """Function to create 2 ImageDataGenerators to split dataset into
train and validation datasets.
    Data augmentation is not implemented for the validation
dataset."""
    height, width = input_shape[:2]
    datagen = ImageDataGenerator(rescale=1.0/255, zoom_range=0.15,
        horizontal_flip=True, vertical_flip=False,
        height_shift_range=0.15, width_shift_range=0.15,
        brightness_range=(0.8, 1.2), rotation_range=20,
        validation_split=validation_split
    )
    train_data = datagen.flow_from_directory(path,
        target_size=(height, width), batch_size=batch_size,
        class_mode='binary', subset='training', seed=seed
    )
    val_datagen = ImageDataGenerator(rescale=1.0/255,
        validation_split=validation_split
    )
    val_data = val_datagen.flow_from_directory(path,
        target_size=(height, width), batch_size=batch_size,
        class_mode='binary', subset='validation', seed=seed
    )
    return train_data, val_data

# Split training and validation datasets
train, val = load_data(path_dataset)

print(f"Classes found: {train.class_indices}")
print(f"Training images: {train.samples}")
print(f"Validation images: {val.samples}")
```

```
Found 19968 images belonging to 2 classes.
Found 4991 images belonging to 2 classes.
Classes found: {'Cat': 0, 'Dog': 1}
Training images: 19968
Validation images: 4991
```

Model retraining (iteration 3)

```
def train_model(model, train_data, val_data, epochs, version_model,
folder_models=folder_models):
    file_name =
os.path.join(folder_models,f'binary_model_v{version_model}.h5')
    callbacks = [
        EarlyStopping(monitor='val_loss', patience=5,
restore_best_weights=True, verbose=0),
        ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=3,
min_lr=1e-6, verbose=0),
        ModelCheckpoint(file_name, monitor='val_accuracy',
save_best_only=True, verbose=1)
    ]

    history = model.fit(train_data, validation_data=val_data,
              epochs=epochs, callbacks=callbacks, verbose=2)

    return model, history

# Load model v2
model_v3 =
load_model(os.path.join(folder_models,'binary_model_v2.h5'))
model_v3.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 resnet50 (Functional)       (None, 7, 7, 2048)        23587712

 global_average_pooling2d (  (None, 2048)              0
 GlobalAveragePooling2D)

 dense (Dense)               (None, 128)               262272

 dense_1 (Dense)             (None, 1)                 129

=================================================================
Total params: 23850113 (90.98 MB)
Trainable params: 9193729 (35.07 MB)
Non-trainable params: 14656384 (55.91 MB)
_________________________________________________________________
```

```
epochs = 20
version_model = 3
print(f"Parameters: batch_size = {batch_size}, learning_rate =
{learning_rate}, neurons = {neurons}, epochs = {epochs}")
```

```
Parameters: batch_size = 8, learning_rate = 1e-05, neurons = 128,
epochs = 20
```

```
# last 20 layers
for layer in model_v3.layers[0].layers[-20:]:
    layer.trainable = True

# Recompile
model_v3.compile(optimizer=Adam(learning_rate=learning_rate),
                 loss='binary_crossentropy', metrics=['accuracy'])
```

```python
print(f"TensorFlow Version: {tf.__version__}")

# Ensure GPU is available
physical_devices = tf.config.list_physical_devices('GPU')
if len(physical_devices) > 0:
    tf.config.experimental.set_memory_growth(physical_devices[0],True)
    print("GPU is available and memory growth is enabled.")
else:
    print("GPU not available, training will be on CPU.")

# Retrain the model
model_v3, history_stage3 = train_model(model_v3, train, val,
epochs=epochs, version_model=version_model)
```

```
TensorFlow Version: 2.13.1
GPU not available, training will be on CPU.
Epoch 1/20

2025-09-29 15:46:37.143909: W
tensorflow/tsl/framework/cpu_allocator_impl.cc:83] Allocation of
25690112 exceeds 10% of free system memory.
2025-09-29 15:46:37.872791: W
tensorflow/tsl/framework/cpu_allocator_impl.cc:83] Allocation of
25690112 exceeds 10% of free system memory.
2025-09-29 15:46:38.004372: W
tensorflow/tsl/framework/cpu_allocator_impl.cc:83] Allocation of
26615808 exceeds 10% of free system memory.
2025-09-29 15:46:38.020688: W
tensorflow/tsl/framework/cpu_allocator_impl.cc:83] Allocation of
25690112 exceeds 10% of free system memory.
2025-09-29 15:46:38.031538: W
tensorflow/tsl/framework/cpu_allocator_impl.cc:83] Allocation of
25690112 exceeds 10% of free system memory.
/home/ant/TensorFlow-Keras-ResNet50-HuggingFace/env/lib/python3.8/
site-packages/PIL/TiffImagePlugin.py:900: UserWarning: Truncated File
Read
  warnings.warn(str(msg))


Epoch 1: val_accuracy improved from -inf to 0.78622, saving model
to ../models/binary_model_v3.h5

/home/ant/TensorFlow-Keras-ResNet50-HuggingFace/env/lib/python3.8/
site-packages/keras/src/engine/training.py:3000: UserWarning: You are
saving your model as an HDF5 file via `model.save()`. This file format
is considered legacy. We recommend using instead the native Keras
format, e.g. `model.save('my_model.keras')`.
  saving_api.save_model(

2496/2496 - 4370s - loss: 0.5271 - accuracy: 0.7321 - val_loss: 0.4475
- val_accuracy: 0.7862 - lr: 1.0000e-05 - 4370s/epoch - 2s/step
```

```
Epoch 2/20

Epoch 2: val_accuracy improved from 0.78622 to 0.78742, saving model
to ../models/binary_model_v3.h5
2496/2496 - 3786s - loss: 0.5264 - accuracy: 0.7339 - val_loss: 0.4528
- val_accuracy: 0.7874 - lr: 1.0000e-05 - 3786s/epoch - 2s/step
Epoch 3/20

Epoch 3: val_accuracy did not improve from 0.78742
2496/2496 - 2353s - loss: 0.5212 - accuracy: 0.7401 - val_loss: 0.4484
- val_accuracy: 0.7812 - lr: 1.0000e-05 - 2353s/epoch - 943ms/step
Epoch 4/20

Epoch 4: val_accuracy improved from 0.78742 to 0.79603, saving model
to ../models/binary_model_v3.h5
2496/2496 - 2674s - loss: 0.5168 - accuracy: 0.7395 - val_loss: 0.4421
- val_accuracy: 0.7960 - lr: 1.0000e-05 - 2674s/epoch - 1s/step
Epoch 5/20

Epoch 5: val_accuracy did not improve from 0.79603
2496/2496 - 4020s - loss: 0.5126 - accuracy: 0.7423 - val_loss: 0.4454
- val_accuracy: 0.7918 - lr: 1.0000e-05 - 4020s/epoch - 2s/step
Epoch 6/20

Epoch 6: val_accuracy did not improve from 0.79603
2496/2496 - 4231s - loss: 0.5125 - accuracy: 0.7456 - val_loss: 0.4539
- val_accuracy: 0.7828 - lr: 1.0000e-05 - 4231s/epoch - 2s/step
Epoch 7/20
```

Result 3: val_accuracy=?%.

```
pd.DataFrame(history_stage3.history).plot(figsize=(12, 4))
plt.show()

# Save model
#
model.save(os.path.join(folder_models,f'binary_model_v{version_model}.
keras'))
```

Finally, the accuracy model is 85%.