# faential_6classes_vggface

June 13, 2023

[1]:

```
!sudo apt-get update
!sudo apt-get install -y gnupg2 curl
!sudo curl -O https://developer.download.nvidia.com/compute/cuda/repos/debian10/
 ↪x86_64/cuda-ubuntu2004.pin
!sudo mv cuda-ubuntu2004.pin /etc/apt/preferences.d/cuda-repository-pin-600
!sudo curl -LO https://developer.download.nvidia.com/compute/cuda/11.4.2/
 ↪local_installers/cuda-repo-debian10-11-4-local_11.4.2-470.57.02-1_amd64.deb
!sudo dpkg -i cuda-repo-debian10-11-4-local_11.4.2-470.57.02-1_amd64.deb
!sudo apt-get update
!sudo apt-get -y install cuda
```

```
Get:1 file:/var/cuda-repo-debian10-11-4-local  InRelease
Ign:1 file:/var/cuda-repo-debian10-11-4-local  InRelease
Hit:2 http://packages.cloud.google.com/apt gcsfuse-bullseye InRelease
Hit:3 http://deb.debian.org/debian bullseye InRelease
Get:4 file:/var/cuda-repo-debian10-11-4-local  Release [564 B]
Get:5 http://deb.debian.org/debian bullseye-updates InRelease [44.1 kB]
Get:4 file:/var/cuda-repo-debian10-11-4-local  Release [564 B]
Hit:6 http://packages.cloud.google.com/apt google-compute-engine-bullseye-stable
InRelease
Hit:7 https://packages.cloud.google.com/apt google-fast-socket InRelease
Get:8 http://security.debian.org/debian-security bullseye-security InRelease
[48.4 kB]
Hit:9 http://packages.cloud.google.com/apt cloud-sdk-bullseye InRelease
Get:10 http://deb.debian.org/debian bullseye-backports InRelease [49.0 kB]
Get:11 https://download.docker.com/linux/debian bullseye InRelease [43.3 kB]
Get:12 file:/var/cuda-repo-debian10-11-4-local  Release.gpg [836 B]
Get:12 file:/var/cuda-repo-debian10-11-4-local  Release.gpg [836 B]
Hit:13 https://nvidia.github.io/libnvidia-container/stable/debian10/amd64
InRelease
Hit:14 https://nvidia.github.io/nvidia-container-runtime/stable/debian10/amd64
InRelease
Hit:15 https://nvidia.github.io/nvidia-docker/debian10/amd64  InRelease
Hit:16 https://packages.cloud.google.com/apt kubernetes-xenial InRelease
Ign:12 file:/var/cuda-repo-debian10-11-4-local  Release.gpg
Reading package lists… Done
W: GPG error: file:/var/cuda-repo-debian10-11-4-local  Release: The following
signatures couldn't be verified because the public key is not available:
```

```
NO_PUBKEY F60F4B3D7FA2AF80
E: The repository 'file:/var/cuda-repo-debian10-11-4-local  Release' is not
signed.
N: Updating from such a repository can't be done securely, and is therefore
disabled by default.
N: See apt-secure(8) manpage for repository creation and user configuration
details.
Reading package lists… Done
E: Unable to parse package file /etc/apt/preferences.d/cuda-repository-pin-600
(1)
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   433  100   433    0     0    391      0  0:00:01  0:00:01 --:--:--   391
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 2432M  100 2432M    0     0   175M      0  0:00:13  0:00:13 --:--:--  183M
(Reading database … 128111 files and directories currently installed.)
Preparing to unpack cuda-repo-debian10-11-4-local_11.4.2-470.57.02-1_amd64.deb
…
Unpacking cuda-repo-debian10-11-4-local (11.4.2-470.57.02-1) over
(11.4.2-470.57.02-1) …
Setting up cuda-repo-debian10-11-4-local (11.4.2-470.57.02-1) …

The public CUDA GPG key does not appear to be installed.
To install the key, run this command:
sudo apt-key add /var/cuda-repo-debian10-11-4-local/7fa2af80.pub

Get:1 file:/var/cuda-repo-debian10-11-4-local  InRelease
Ign:1 file:/var/cuda-repo-debian10-11-4-local  InRelease
Get:2 file:/var/cuda-repo-debian10-11-4-local  Release [564 B]
Get:2 file:/var/cuda-repo-debian10-11-4-local  Release [564 B]
Get:3 file:/var/cuda-repo-debian10-11-4-local  Release.gpg [836 B]
Get:3 file:/var/cuda-repo-debian10-11-4-local  Release.gpg [836 B]
Hit:4 http://deb.debian.org/debian bullseye InRelease
Hit:5 http://packages.cloud.google.com/apt gcsfuse-bullseye InRelease
Hit:6 http://security.debian.org/debian-security bullseye-security InRelease
Hit:7 https://packages.cloud.google.com/apt google-fast-socket InRelease
Hit:8 http://packages.cloud.google.com/apt google-compute-engine-bullseye-stable
InRelease
Get:9 https://download.docker.com/linux/debian bullseye InRelease [43.3 kB]
Hit:10 http://packages.cloud.google.com/apt cloud-sdk-bullseye InRelease
Ign:3 file:/var/cuda-repo-debian10-11-4-local  Release.gpg
Hit:11 http://deb.debian.org/debian bullseye-updates InRelease
Hit:12 http://deb.debian.org/debian bullseye-backports InRelease
Hit:13 https://nvidia.github.io/libnvidia-container/stable/debian10/amd64
InRelease
Hit:14 https://packages.cloud.google.com/apt kubernetes-xenial InRelease
Hit:15 https://nvidia.github.io/nvidia-container-runtime/stable/debian10/amd64
```

```
InRelease
Hit:16 https://nvidia.github.io/nvidia-docker/debian10/amd64  InRelease
Reading package lists… Done
W: GPG error: file:/var/cuda-repo-debian10-11-4-local  Release: The following
signatures couldn't be verified because the public key is not available:
NO_PUBKEY F60F4B3D7FA2AF80
E: The repository 'file:/var/cuda-repo-debian10-11-4-local  Release' is not
signed.
N: Updating from such a repository can't be done securely, and is therefore
disabled by default.
N: See apt-secure(8) manpage for repository creation and user configuration
details.
Reading package lists… Done
E: Unable to parse package file /etc/apt/preferences.d/cuda-repository-pin-600
(1)
```

[2]:

```
!nvidia-smi
```

```
Tue Jun 13 11:25:31 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 510.47.03    Driver Version: 510.47.03    CUDA Version: 11.6     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla T4            Off  | 00000000:00:04.0 Off |                    0 |
| N/A   68C    P0    33W /  70W |      0MiB / 15360MiB |     11%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

[3]:

```
!pip install --user tensorflow==2.11.1
```

```
WARNING: Ignoring invalid distribution -eras
(/opt/conda/lib/python3.10/site-packages)
WARNING: Ignoring invalid distribution -rapt
(/opt/conda/lib/python3.10/site-packages)
Requirement already satisfied: tensorflow==2.11.1 in
/home/jupyter/.local/lib/python3.10/site-packages (2.11.1)
```

```
Requirement already satisfied: absl-py>=1.0.0 in /opt/conda/lib/python3.10/site-
packages (from tensorflow==2.11.1) (1.4.0)
Requirement already satisfied: astunparse>=1.6.0 in
/opt/conda/lib/python3.10/site-packages (from tensorflow==2.11.1) (1.6.3)
Requirement already satisfied: flatbuffers>=2.0 in
/opt/conda/lib/python3.10/site-packages (from tensorflow==2.11.1) (23.3.3)
Requirement already satisfied: gast<=0.4.0,>=0.2.1 in
/opt/conda/lib/python3.10/site-packages (from tensorflow==2.11.1) (0.4.0)
Requirement already satisfied: google-pasta>=0.1.1 in
/opt/conda/lib/python3.10/site-packages (from tensorflow==2.11.1) (0.2.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in
/home/jupyter/.local/lib/python3.10/site-packages (from tensorflow==2.11.1)
(1.54.2)
Requirement already satisfied: h5py>=2.9.0 in /opt/conda/lib/python3.10/site-
packages (from tensorflow==2.11.1) (3.8.0)
Requirement already satisfied: keras<2.12,>=2.11.0 in
/home/jupyter/.local/lib/python3.10/site-packages (from tensorflow==2.11.1)
(2.11.0)
Requirement already satisfied: libclang>=13.0.0 in
/opt/conda/lib/python3.10/site-packages (from tensorflow==2.11.1) (16.0.0)
Requirement already satisfied: numpy>=1.20 in /opt/conda/lib/python3.10/site-
packages (from tensorflow==2.11.1) (1.23.5)
Requirement already satisfied: opt-einsum>=2.3.2 in
/opt/conda/lib/python3.10/site-packages (from tensorflow==2.11.1) (3.3.0)
Requirement already satisfied: packaging in
/home/jupyter/.local/lib/python3.10/site-packages (from tensorflow==2.11.1)
(20.9)
Requirement already satisfied: protobuf<3.20,>=3.9.2 in
/home/jupyter/.local/lib/python3.10/site-packages (from tensorflow==2.11.1)
(3.19.6)
Requirement already satisfied: setuptools in /opt/conda/lib/python3.10/site-
packages (from tensorflow==2.11.1) (67.7.2)
Requirement already satisfied: six>=1.12.0 in /opt/conda/lib/python3.10/site-
packages (from tensorflow==2.11.1) (1.16.0)
Requirement already satisfied: tensorboard<2.12,>=2.11 in
/home/jupyter/.local/lib/python3.10/site-packages (from tensorflow==2.11.1)
(2.11.2)
Requirement already satisfied: tensorflow-estimator<2.12,>=2.11.0 in
/home/jupyter/.local/lib/python3.10/site-packages (from tensorflow==2.11.1)
(2.11.0)
Requirement already satisfied: termcolor>=1.1.0 in
/opt/conda/lib/python3.10/site-packages (from tensorflow==2.11.1) (2.3.0)
Requirement already satisfied: typing-extensions>=3.6.6 in
/opt/conda/lib/python3.10/site-packages (from tensorflow==2.11.1) (4.5.0)
Requirement already satisfied: wrapt>=1.11.0 in /opt/conda/lib/python3.10/site-
packages (from tensorflow==2.11.1) (1.14.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in
/opt/conda/lib/python3.10/site-packages (from tensorflow==2.11.1) (0.29.0)
```

```
Requirement already satisfied: wheel<1.0,>=0.23.0 in
/opt/conda/lib/python3.10/site-packages (from
astunparse>=1.6.0->tensorflow==2.11.1) (0.40.0)
Requirement already satisfied: google-auth<3,>=1.6.3 in
/opt/conda/lib/python3.10/site-packages (from
tensorboard<2.12,>=2.11->tensorflow==2.11.1) (2.17.3)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in
/home/jupyter/.local/lib/python3.10/site-packages (from
tensorboard<2.12,>=2.11->tensorflow==2.11.1) (0.4.6)
Requirement already satisfied: markdown>=2.6.8 in
/opt/conda/lib/python3.10/site-packages (from
tensorboard<2.12,>=2.11->tensorflow==2.11.1) (3.4.3)
Requirement already satisfied: requests<3,>=2.21.0 in
/opt/conda/lib/python3.10/site-packages (from
tensorboard<2.12,>=2.11->tensorflow==2.11.1) (2.28.2)
Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0 in
/home/jupyter/.local/lib/python3.10/site-packages (from
tensorboard<2.12,>=2.11->tensorflow==2.11.1) (0.6.1)
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in
/opt/conda/lib/python3.10/site-packages (from
tensorboard<2.12,>=2.11->tensorflow==2.11.1) (1.8.1)
Requirement already satisfied: werkzeug>=1.0.1 in
/opt/conda/lib/python3.10/site-packages (from
tensorboard<2.12,>=2.11->tensorflow==2.11.1) (2.1.2)
Requirement already satisfied: pyparsing>=2.0.2 in
/opt/conda/lib/python3.10/site-packages (from packaging->tensorflow==2.11.1)
(3.0.9)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in
/opt/conda/lib/python3.10/site-packages (from google-
auth<3,>=1.6.3->tensorboard<2.12,>=2.11->tensorflow==2.11.1) (4.2.4)
Requirement already satisfied: pyasn1-modules>=0.2.1 in
/opt/conda/lib/python3.10/site-packages (from google-
auth<3,>=1.6.3->tensorboard<2.12,>=2.11->tensorflow==2.11.1) (0.2.7)
Requirement already satisfied: rsa<5,>=3.1.4 in /opt/conda/lib/python3.10/site-
packages (from google-
auth<3,>=1.6.3->tensorboard<2.12,>=2.11->tensorflow==2.11.1) (4.9)
Requirement already satisfied: requests-oauthlib>=0.7.0 in
/opt/conda/lib/python3.10/site-packages (from google-auth-
oauthlib<0.5,>=0.4.1->tensorboard<2.12,>=2.11->tensorflow==2.11.1) (1.3.1)
Requirement already satisfied: charset-normalizer<4,>=2 in
/opt/conda/lib/python3.10/site-packages (from
requests<3,>=2.21.0->tensorboard<2.12,>=2.11->tensorflow==2.11.1) (2.1.1)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/lib/python3.10/site-
packages (from requests<3,>=2.21.0->tensorboard<2.12,>=2.11->tensorflow==2.11.1)
(3.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
/opt/conda/lib/python3.10/site-packages (from
requests<3,>=2.21.0->tensorboard<2.12,>=2.11->tensorflow==2.11.1) (1.26.15)
```

```
Requirement already satisfied: certifi>=2017.4.17 in
/opt/conda/lib/python3.10/site-packages (from
requests<3,>=2.21.0->tensorboard<2.12,>=2.11->tensorflow==2.11.1) (2022.12.7)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in
/opt/conda/lib/python3.10/site-packages (from pyasn1-modules>=0.2.1->google-
auth<3,>=1.6.3->tensorboard<2.12,>=2.11->tensorflow==2.11.1) (0.4.8)
Requirement already satisfied: oauthlib>=3.0.0 in
/opt/conda/lib/python3.10/site-packages (from requests-oauthlib>=0.7.0->google-
auth-oauthlib<0.5,>=0.4.1->tensorboard<2.12,>=2.11->tensorflow==2.11.1) (3.2.2)
WARNING: Ignoring invalid distribution -eras

(/opt/conda/lib/python3.10/site-packages)

WARNING: Ignoring invalid distribution -rapt

(/opt/conda/lib/python3.10/site-packages)
```

[4]:

```python
import tensorflow as tf
```

```
2023-06-13 11:25:39.221394: I tensorflow/core/platform/cpu_feature_guard.cc:193]
This TensorFlow binary is optimized with oneAPI Deep Neural Network Library
(oneDNN) to use the following CPU instructions in performance-critical
operations:  AVX2 AVX512F FMA
To enable them in other operations, rebuild TensorFlow with the appropriate
compiler flags.
2023-06-13 11:25:53.222188: W
tensorflow/compiler/xla/stream_executor/platform/default/dso_loader.cc:64] Could
not load dynamic library 'libnvinfer.so.7'; dlerror: libnvinfer.so.7: cannot
open shared object file: No such file or directory; LD_LIBRARY_PATH:
/usr/local/cuda/lib64:/usr/local/nccl2/lib:/usr/local/cuda/extras/CUPTI/lib64
2023-06-13 11:25:53.222340: W
tensorflow/compiler/xla/stream_executor/platform/default/dso_loader.cc:64] Could
not load dynamic library 'libnvinfer_plugin.so.7'; dlerror:
libnvinfer_plugin.so.7: cannot open shared object file: No such file or
directory; LD_LIBRARY_PATH:
/usr/local/cuda/lib64:/usr/local/nccl2/lib:/usr/local/cuda/extras/CUPTI/lib64
2023-06-13 11:25:53.222351: W
tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Cannot
dlopen some TensorRT libraries. If you would like to use Nvidia GPU with
TensorRT, please make sure the missing libraries mentioned above are installed
properly.
```

[5]:

```python
print(tf.__version__)
```

```
2.11.1
```

[6]:

```python
from keras.layers import Input, Lambda, Dense, Flatten
from keras.models import Model
from keras.preprocessing import image
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
import numpy as np
from glob import glob
import matplotlib.pyplot as plt
```

[7]:

```
import os
import zipfile
```

define the directories containing your images

[8]:

```
# variabel directory for training pict for each category
train_oily_dir = os.path.join("/home/jupyter/content/faceSkin_tipe_train/oily")
train_normal_dir = os.path.join("/home/jupyter/content/faceSkin_tipe_train/
  ↪normal")
train_combination_dir =os.path.join("/home/jupyter/content/faceSkin_tipe_train/
  ↪combination")
train_sensitive_dir = os.path.join("/home/jupyter/content/faceSkin_tipe_train/
  ↪sensitive")
train_dry_dir = os.path.join("/home/jupyter/content/faceSkin_tipe_train/dry")
train_nonface_dir = os.path.join("/home/jupyter/content/faceSkin_tipe_train/
  ↪nonface/")

# variabel directory for validation pict for each category
validation_oily_dir =os.path.join("/home/jupyter/content/
  ↪faceSkin_tipe_validation/oily")
validation_normal_dir =os.path.join("/home/jupyter/content/
  ↪faceSkin_tipe_validation/normal")
validation_combination_dir =os.path.join("/home/jupyter/content/
  ↪faceSkin_tipe_validation/combination")
validation_sensitive_dir =os.path.join("/home/jupyter/content/
  ↪faceSkin_tipe_validation/sensitive")
validation_dry_dir =os.path.join("/home/jupyter/content/
  ↪faceSkin_tipe_validation/dry")
validation_nonface_dir =os.path.join("/home/jupyter/content/
  ↪faceSkin_tipe_validation/nonface")
```

filenames check in directory

[9]:

```
train_dry_names = os.listdir(train_dry_dir)
train_oily_names = os.listdir(train_oily_dir)
train_normal_names =  os.listdir(train_normal_dir)
train_combination_names =  os.listdir(train_combination_dir)
train_sensitive_names =  os.listdir(train_sensitive_dir)
train_nonface_names =  os.listdir(train_nonface_dir)
```

```python
validation_dry_names = os.listdir(validation_dry_dir)
validation_oily_names = os.listdir(validation_oily_dir)
validation_normal_names =  os.listdir(validation_normal_dir)
validation_combination_names =  os.listdir(validation_combination_dir)
validation_sensitive_names =  os.listdir(validation_sensitive_dir)
validation_nonface_names =  os.listdir(validation_nonface_dir)

print(f'TRAIN SET DRY: {train_dry_names[:5]}')
print(f'TRAIN SET OILY: {train_dry_names[:5]}')
print(f'TRAIN SET SENSITIVE: {train_dry_names[:5]}')
print(f'TRAIN SET COMBINATION: {train_dry_names[:5]}')
print(f'TRAIN SET NORMAL: {train_dry_names[:5]}')
print(f'TRAIN SET NONFACE: {train_nonface_names[:5]} \n')

print(f'VALIDATION SET DRY: {validation_dry_names[:5]}')
print(f'VALIDATION SET OILY: {validation_dry_names[:5]}')
print(f'VALIDATION SET SENSITIVE: {validation_dry_names[:5]}')
print(f'VALIDATION SET COMBINATION: {validation_dry_names[:5]}')
print(f'VALIDATION SET NORMAL: {validation_dry_names[:5]}')
print(f'VALIDATION SET NONFACE: {validation_nonface_names[:5]}')
```

```
TRAIN SET DRY: ['f1-006-01.jpg', 'folder(185)3.jpg', 'folder(95)2.jpg',
'folder(224)2.jpg', 'folder(219)2.jpg']
TRAIN SET OILY: ['f1-006-01.jpg', 'folder(185)3.jpg', 'folder(95)2.jpg',
'folder(224)2.jpg', 'folder(219)2.jpg']
TRAIN SET SENSITIVE: ['f1-006-01.jpg', 'folder(185)3.jpg', 'folder(95)2.jpg',
'folder(224)2.jpg', 'folder(219)2.jpg']
TRAIN SET COMBINATION: ['f1-006-01.jpg', 'folder(185)3.jpg', 'folder(95)2.jpg',
'folder(224)2.jpg', 'folder(219)2.jpg']
TRAIN SET NORMAL: ['f1-006-01.jpg', 'folder(185)3.jpg', 'folder(95)2.jpg',
'folder(224)2.jpg', 'folder(219)2.jpg']
TRAIN SET NONFACE: ['000000013201.jpg', '000000017115.jpg', '000000002473.jpg',
'000000024021.jpg', '000000015335.jpg']

VALIDATION SET DRY: ['folder(7)2.jpg', 'dd.jpg', 'folder(72)3.jpg',
'folder(77)3.jpg', '1 (72).jpg']
VALIDATION SET OILY: ['folder(7)2.jpg', 'dd.jpg', 'folder(72)3.jpg',
'folder(77)3.jpg', '1 (72).jpg']
VALIDATION SET SENSITIVE: ['folder(7)2.jpg', 'dd.jpg', 'folder(72)3.jpg',
'folder(77)3.jpg', '1 (72).jpg']
VALIDATION SET COMBINATION: ['folder(7)2.jpg', 'dd.jpg', 'folder(72)3.jpg',
'folder(77)3.jpg', '1 (72).jpg']
VALIDATION SET NORMAL: ['folder(7)2.jpg', 'dd.jpg', 'folder(72)3.jpg',
'folder(77)3.jpg', '1 (72).jpg']
VALIDATION SET NONFACE: ['000000051008.jpg', '000000051976.jpg',
'000000045596.jpg', '000000047010.jpg', '000000045090.jpg']
```

chacking total number of images for each categories in training and validation directories

[10]:
```
print(f'total training oily images: {len(os.listdir(train_oily_dir))}')
print(f'total training dry images: {len(os.listdir(train_dry_dir))}')
print(f'total training normal images: {len(os.listdir(train_normal_dir))}')
print(f'total training combination images: {len(os.
 ↪listdir(train_combination_dir))}')
print(f'total training sensitive images: {len(os.
 ↪listdir(train_sensitive_dir))}')
print(f'total training nonface images: {len(os.listdir(train_nonface_dir))}\n')


print(f'total validation oily images: {len(os.listdir(validation_oily_dir))}')
print(f'total validation dry images: {len(os.listdir(validation_dry_dir))}')
print(f'total validation normal images: {len(os.
 ↪listdir(validation_normal_dir))}')
print(f'total validation combination images: {len(os.
 ↪listdir(validation_combination_dir))}')
print(f'total validation sensitive images: {len(os.
 ↪listdir(validation_sensitive_dir))}')
print(f'total validation nonface images: {len(os.
 ↪listdir(validation_nonface_dir))}')
```

```
total training oily images: 209
total training dry images: 207
total training normal images: 99
total training combination images: 110
total training sensitive images: 202
total training nonface images: 200

total validation oily images: 53
total validation dry images: 50
total validation normal images: 25
total validation combination images: 36
total validation sensitive images: 49
total validation nonface images: 50
```

[ ]:
```

```

## 1 Data preprocessing

using image data generator

[29]:
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import os

# All images will be rescaled by 1./255
```

```
train_datagen = ImageDataGenerator(
      rescale=1/255,
      rotation_range=40,
      width_shift_range=0.2,
      height_shift_range=0.2,
      shear_range=0.2,
      zoom_range=0.2,
      horizontal_flip=True,
      fill_mode='nearest')

validation_datagen = ImageDataGenerator(rescale=1/255)

# Set the base directory where your data is located
base_directory = '/home/jupyter/content/'

# Get the list of subdirectories (classes) in the training directory
train_classes = [subdir for subdir in os.listdir(os.path.join(base_directory,␣
 ↪'faceSkin_tipe_train')) if os.path.isdir(os.path.join(base_directory,␣
 ↪'faceSkin_tipe_train', subdir))]

# Remove the ".ipynb_checkpoints" class from the list if it exists
if '.ipynb_checkpoints' in train_classes:
    train_classes.remove('.ipynb_checkpoints')

# Flow training images in batches of 128 using train_datagen generator
train_generator = train_datagen.flow_from_directory(
        os.path.join(base_directory, 'faceSkin_tipe_train'),  # This is the␣
 ↪source directory for training images
        target_size=(224, 224),  # All images will be resized to 224x224
        batch_size=32,
        class_mode='categorical',
        classes=train_classes)

# Get the list of subdirectories (classes) in the validation directory
validation_classes = [subdir for subdir in os.listdir(os.path.
 ↪join(base_directory, 'faceSkin_tipe_validation')) if os.path.isdir(os.path.
 ↪join(base_directory, 'faceSkin_tipe_validation', subdir))]

# Remove the ".ipynb_checkpoints" class from the list if it exists
if '.ipynb_checkpoints' in validation_classes:
    validation_classes.remove('.ipynb_checkpoints')

# Flow validation images in batches of 128 using validation_datagen generator
validation_generator = validation_datagen.flow_from_directory(
        os.path.join(base_directory, 'faceSkin_tipe_validation'),  # This is␣
 ↪the source directory for validation images
        target_size=(224, 224),  # All images will be resized to 224x224
        batch_size=8,
        class_mode='categorical',
        classes=validation_classes)


# Get the class names from the generator's class_indices dictionary
class_names = list(train_generator.class_indices.keys())

# Print the class names
print("Class Names:", class_names)
```

```
Found 1027 images belonging to 6 classes.
Found 263 images belonging to 6 classes.
Class Names: ['oily', 'normal', 'sensitive', 'dry', 'nonface', 'combination']
```

[32]:

```
# Access the class indices
class_indices = train_generator.class_indices

# Print the list of classes
print("List of Classes:")
for class_name, class_index in class_indices.items():
    print(class_name, ":", class_index)
```

```
List of Classes:
oily : 0
normal : 1
sensitive : 2
dry : 3
nonface : 4
combination : 5
```

load pretrained model

[33]:

```
!python --version
```

```
Python 3.10.10
```

[14]:

```
!pip install keras_vggface
```

```
WARNING: Ignoring invalid distribution -eras

(/opt/conda/lib/python3.10/site-packages)

WARNING: Ignoring invalid distribution -rapt

(/opt/conda/lib/python3.10/site-packages)

Requirement already satisfied: keras_vggface in
/opt/conda/lib/python3.10/site-packages (0.6)
Requirement already satisfied: numpy>=1.9.1 in /opt/conda/lib/python3.10/site-
packages (from keras_vggface) (1.23.5)
Requirement already satisfied: scipy>=0.14 in /opt/conda/lib/python3.10/site-
```

```
packages (from keras_vggface) (1.9.3)
Requirement already satisfied: h5py in /opt/conda/lib/python3.10/site-packages
(from keras_vggface) (3.8.0)
Requirement already satisfied: pillow in /opt/conda/lib/python3.10/site-packages
(from keras_vggface) (9.5.0)
Requirement already satisfied: keras in
/home/jupyter/.local/lib/python3.10/site-packages (from keras_vggface) (2.11.0)
Requirement already satisfied: six>=1.9.0 in /opt/conda/lib/python3.10/site-
packages (from keras_vggface) (1.16.0)
Requirement already satisfied: pyyaml in /opt/conda/lib/python3.10/site-packages
(from keras_vggface) (5.4.1)
WARNING: Ignoring invalid distribution -eras
(/opt/conda/lib/python3.10/site-packages)
WARNING: Ignoring invalid distribution -rapt
(/opt/conda/lib/python3.10/site-packages)
```

[15]:

```
!pip install keras_applications
```

```
WARNING: Ignoring invalid distribution -eras
(/opt/conda/lib/python3.10/site-packages)
WARNING: Ignoring invalid distribution -rapt
(/opt/conda/lib/python3.10/site-packages)
Requirement already satisfied: keras_applications in
/opt/conda/lib/python3.10/site-packages (1.0.8)
Requirement already satisfied: numpy>=1.9.1 in /opt/conda/lib/python3.10/site-
packages (from keras_applications) (1.23.5)
Requirement already satisfied: h5py in /opt/conda/lib/python3.10/site-packages
(from keras_applications) (3.8.0)
WARNING: Ignoring invalid distribution -eras
(/opt/conda/lib/python3.10/site-packages)
WARNING: Ignoring invalid distribution -rapt
(/opt/conda/lib/python3.10/site-packages)
```

[16]:

```
!pip show keras
```

```
WARNING: Ignoring invalid distribution -eras
(/opt/conda/lib/python3.10/site-packages)
WARNING: Ignoring invalid distribution -rapt
(/opt/conda/lib/python3.10/site-packages)
Name: keras
Version: 2.11.0
Summary: Deep learning for humans.
Home-page: https://keras.io/
Author: Keras team
Author-email: keras-users@googlegroups.com
License: Apache 2.0
Location: /home/jupyter/.local/lib/python3.10/site-packages
Requires:
Required-by: keras-vggface, tensorflow
```

[34]:

```python
import tensorflow.keras as keras
```

[35]:

```python
print(tf.__version__)
```

```
2.11.1
```

[36]:

```python
from keras_vggface.vggface import VGGFace
from tensorflow.keras import layers

# Set the weights file you downloaded into a variable
local_weights_file = '/home/jupyter/content/vgg_face_weights.h5'

# Initialize the base model.
# Set the input shape and remove the dense layers.
pre_trained_model = VGGFace(input_shape = (224, 224, 3),
                                include_top = False,
                                model = 'vgg16')

# Load the pre-trained weights you downloaded.
pre_trained_model.load_weights(local_weights_file, by_name=True,␣
 ↪skip_mismatch=True)

# Freeze the weights of the layers.
for layer in pre_trained_model.layers:
  layer.trainable = False
```

[37]:

```python
pre_trained_model.summary()
```

```
Model: "vggface_vgg16"

_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_3 (InputLayer)        [(None, 224, 224, 3)]     0

 conv1_1 (Conv2D)            (None, 224, 224, 64)      1792

 conv1_2 (Conv2D)            (None, 224, 224, 64)      36928

 pool1 (MaxPooling2D)        (None, 112, 112, 64)      0

 conv2_1 (Conv2D)            (None, 112, 112, 128)     73856

 conv2_2 (Conv2D)            (None, 112, 112, 128)     147584

 pool2 (MaxPooling2D)        (None, 56, 56, 128)       0

 conv3_1 (Conv2D)            (None, 56, 56, 256)       295168

 conv3_2 (Conv2D)            (None, 56, 56, 256)       590080

 conv3_3 (Conv2D)            (None, 56, 56, 256)       590080

 pool3 (MaxPooling2D)        (None, 28, 28, 256)       0

 conv4_1 (Conv2D)            (None, 28, 28, 512)       1180160

 conv4_2 (Conv2D)            (None, 28, 28, 512)       2359808

 conv4_3 (Conv2D)            (None, 28, 28, 512)       2359808

 pool4 (MaxPooling2D)        (None, 14, 14, 512)       0

 conv5_1 (Conv2D)            (None, 14, 14, 512)       2359808

 conv5_2 (Conv2D)            (None, 14, 14, 512)       2359808

 conv5_3 (Conv2D)            (None, 14, 14, 512)       2359808

 pool5 (MaxPooling2D)        (None, 7, 7, 512)         0

=================================================================
Total params: 14,714,688
Trainable params: 0
Non-trainable params: 14,714,688
_________________________________________________________________
```

[39]:
```
# Choose `mixed_7` as the last layer of your base model
last_layer = pre_trained_model.get_layer('conv5_3')
print('last layer output shape: ', last_layer.output_shape)
```

```
last_output = last_layer.output
print('last layer output: ', last_output)
```

```
last layer output shape:  (None, 14, 14, 512)
last layer output:  KerasTensor(type_spec=TensorSpec(shape=(None, 14, 14, 512),
dtype=tf.float32, name=None), name='conv5_3/Relu:0', description="created by
layer 'conv5_3'")
```

## 2 add dense layer depends on the classification (5 category)

[40]:

```
from tensorflow.keras.optimizers import RMSprop
from tensorflow.keras import Model


# Flatten the output layer to 1 dimension
x = layers.Flatten()(last_output)
# Add a fully connected layer with 1,024 hidden units and ReLU activation
x = layers.Dense(1024, activation="ReLU")(x)
# Add a dropout rate of 0.2
x = layers.Dropout(0.2)(x)
# Add a final sigmoid layer for classification
x = layers.Dense(6, activation='softmax')(x)

# Append the dense network to the base model
model = Model(pre_trained_model.input, x)

# Print the model summary. See your dense network connected at the end.
model.summary()
```

```
Model: "model_1"

_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_3 (InputLayer)        [(None, 224, 224, 3)]     0

 conv1_1 (Conv2D)            (None, 224, 224, 64)      1792

 conv1_2 (Conv2D)            (None, 224, 224, 64)      36928

 pool1 (MaxPooling2D)        (None, 112, 112, 64)      0

 conv2_1 (Conv2D)            (None, 112, 112, 128)     73856

 conv2_2 (Conv2D)            (None, 112, 112, 128)     147584

 pool2 (MaxPooling2D)        (None, 56, 56, 128)       0
 conv3_1 (Conv2D)            (None, 56, 56, 256)       295168

 conv3_2 (Conv2D)            (None, 56, 56, 256)       590080

 conv3_3 (Conv2D)            (None, 56, 56, 256)       590080

 pool3 (MaxPooling2D)        (None, 28, 28, 256)       0

 conv4_1 (Conv2D)            (None, 28, 28, 512)       1180160

 conv4_2 (Conv2D)            (None, 28, 28, 512)       2359808

 conv4_3 (Conv2D)            (None, 28, 28, 512)       2359808

 pool4 (MaxPooling2D)        (None, 14, 14, 512)       0

 conv5_1 (Conv2D)            (None, 14, 14, 512)       2359808

 conv5_2 (Conv2D)            (None, 14, 14, 512)       2359808

 conv5_3 (Conv2D)            (None, 14, 14, 512)       2359808

 flatten_1 (Flatten)         (None, 100352)            0

 dense_2 (Dense)             (None, 1024)              102761472

 dropout_1 (Dropout)         (None, 1024)              0

 dense_3 (Dense)             (None, 6)                 6150

=================================================================
Total params: 117,482,310
Trainable params: 102,767,622
Non-trainable params: 14,714,688
_________________________________________________________________
```

compiling the model

```
[41]: from tensorflow.keras.optimizers import Adam
      model.compile(loss=tf.keras.losses.CategoricalCrossentropy(),
                    optimizer=tf.keras.optimizers.Adam(learning_rate = 0.0001),
                    metrics=['accuracy'])

      #membuat callback untuk menentukan learning rate terbaik
      #lr_scheduler = tf.keras.callbacks.LearningRateScheduler(lambda epoch:1e-4 * 10␣
       ↪**(epoch/20))
```

Training

[42]:
```
len(validation_generator)
```

[42]:
```
33
```

[43]:
```
history = model.fit(
      train_generator,
      steps_per_epoch=len(train_generator),
      epochs=150,
      #callbacks = [lr_scheduler],
      verbose=2,
      validation_data = validation_generator,
      validation_steps=len(validation_generator))
```

```
Epoch 1/150
33/33 - 70s - loss: 1.4398 - accuracy: 0.4109 - val_loss: 1.2226 - val_accuracy:
0.5095 - 70s/epoch - 2s/step
Epoch 2/150
33/33 - 67s - loss: 1.1548 - accuracy: 0.5161 - val_loss: 1.1558 - val_accuracy:
0.5057 - 67s/epoch - 2s/step
Epoch 3/150
33/33 - 66s - loss: 1.0851 - accuracy: 0.5599 - val_loss: 1.1255 - val_accuracy:
0.5589 - 66s/epoch - 2s/step
Epoch 4/150
33/33 - 66s - loss: 1.0252 - accuracy: 0.5774 - val_loss: 1.1095 - val_accuracy:
0.5285 - 66s/epoch - 2s/step
Epoch 5/150
33/33 - 66s - loss: 0.9868 - accuracy: 0.5930 - val_loss: 1.0605 - val_accuracy:
0.5589 - 66s/epoch - 2s/step
Epoch 6/150
33/33 - 66s - loss: 0.9692 - accuracy: 0.6144 - val_loss: 1.0511 - val_accuracy:
0.5817 - 66s/epoch - 2s/step
Epoch 7/150
33/33 - 67s - loss: 0.9516 - accuracy: 0.6203 - val_loss: 1.1123 - val_accuracy:
0.5589 - 67s/epoch - 2s/step
Epoch 8/150
33/33 - 66s - loss: 0.9241 - accuracy: 0.6426 - val_loss: 1.0725 - val_accuracy:
0.5665 - 66s/epoch - 2s/step
Epoch 9/150
33/33 - 66s - loss: 0.8538 - accuracy: 0.6670 - val_loss: 1.0316 - val_accuracy:
0.5665 - 66s/epoch - 2s/step
Epoch 10/150
33/33 - 67s - loss: 0.8768 - accuracy: 0.6553 - val_loss: 1.0209 - val_accuracy:
0.5932 - 67s/epoch - 2s/step
Epoch 11/150
33/33 - 67s - loss: 0.8487 - accuracy: 0.6816 - val_loss: 1.0792 - val_accuracy:
0.5817 - 67s/epoch - 2s/step
Epoch 12/150
33/33 - 66s - loss: 0.8309 - accuracy: 0.6758 - val_loss: 1.0211 - val_accuracy:
```

```
0.5856 - 66s/epoch - 2s/step
Epoch 13/150
33/33 - 66s - loss: 0.8257 - accuracy: 0.6777 - val_loss: 0.9691 - val_accuracy:
0.6464 - 66s/epoch - 2s/step
Epoch 14/150
33/33 - 65s - loss: 0.8078 - accuracy: 0.6991 - val_loss: 1.0090 - val_accuracy:
0.6274 - 65s/epoch - 2s/step
Epoch 15/150
33/33 - 65s - loss: 0.7967 - accuracy: 0.6952 - val_loss: 0.9794 - val_accuracy:
0.6236 - 65s/epoch - 2s/step
Epoch 16/150
33/33 - 69s - loss: 0.7837 - accuracy: 0.6933 - val_loss: 0.9584 - val_accuracy:
0.6046 - 69s/epoch - 2s/step
Epoch 17/150
33/33 - 66s - loss: 0.7665 - accuracy: 0.6884 - val_loss: 1.0217 - val_accuracy:
0.6122 - 66s/epoch - 2s/step
Epoch 18/150
33/33 - 66s - loss: 0.7563 - accuracy: 0.7147 - val_loss: 1.0380 - val_accuracy:
0.5970 - 66s/epoch - 2s/step
Epoch 19/150
33/33 - 67s - loss: 0.7777 - accuracy: 0.7020 - val_loss: 1.0630 - val_accuracy:
0.6084 - 67s/epoch - 2s/step
Epoch 20/150
33/33 - 68s - loss: 0.7531 - accuracy: 0.7001 - val_loss: 0.9988 - val_accuracy:
0.6274 - 68s/epoch - 2s/step
Epoch 21/150
33/33 - 67s - loss: 0.7523 - accuracy: 0.7128 - val_loss: 0.9812 - val_accuracy:
0.6008 - 67s/epoch - 2s/step
Epoch 22/150
33/33 - 68s - loss: 0.7268 - accuracy: 0.7167 - val_loss: 1.0423 - val_accuracy:
0.6312 - 68s/epoch - 2s/step
Epoch 23/150
33/33 - 65s - loss: 0.7282 - accuracy: 0.7303 - val_loss: 0.9743 - val_accuracy:
0.6122 - 65s/epoch - 2s/step
Epoch 24/150
33/33 - 66s - loss: 0.6961 - accuracy: 0.7352 - val_loss: 0.9411 - val_accuracy:
0.6160 - 66s/epoch - 2s/step
Epoch 25/150
33/33 - 66s - loss: 0.6895 - accuracy: 0.7342 - val_loss: 1.0155 - val_accuracy:
0.6198 - 66s/epoch - 2s/step
Epoch 26/150
33/33 - 65s - loss: 0.6883 - accuracy: 0.7381 - val_loss: 0.9566 - val_accuracy:
0.6084 - 65s/epoch - 2s/step
Epoch 27/150
33/33 - 65s - loss: 0.6631 - accuracy: 0.7585 - val_loss: 0.9762 - val_accuracy:
0.6160 - 65s/epoch - 2s/step
Epoch 28/150
33/33 - 66s - loss: 0.6373 - accuracy: 0.7673 - val_loss: 1.0081 - val_accuracy:
```

```
0.6046 - 66s/epoch - 2s/step
Epoch 29/150
33/33 - 65s - loss: 0.6826 - accuracy: 0.7303 - val_loss: 0.9875 - val_accuracy:
0.6274 - 65s/epoch - 2s/step
Epoch 30/150
33/33 - 66s - loss: 0.6540 - accuracy: 0.7702 - val_loss: 0.9867 - val_accuracy:
0.6008 - 66s/epoch - 2s/step
Epoch 31/150
33/33 - 65s - loss: 0.6468 - accuracy: 0.7605 - val_loss: 0.9924 - val_accuracy:
0.6160 - 65s/epoch - 2s/step
Epoch 32/150
33/33 - 64s - loss: 0.6205 - accuracy: 0.7692 - val_loss: 1.0685 - val_accuracy:
0.6426 - 64s/epoch - 2s/step
Epoch 33/150
33/33 - 64s - loss: 0.6299 - accuracy: 0.7731 - val_loss: 1.0772 - val_accuracy:
0.6122 - 64s/epoch - 2s/step
Epoch 34/150
33/33 - 64s - loss: 0.6296 - accuracy: 0.7527 - val_loss: 0.9748 - val_accuracy:
0.6122 - 64s/epoch - 2s/step
Epoch 35/150
33/33 - 64s - loss: 0.6051 - accuracy: 0.7760 - val_loss: 0.9690 - val_accuracy:
0.6274 - 64s/epoch - 2s/step
Epoch 36/150
33/33 - 65s - loss: 0.6188 - accuracy: 0.7722 - val_loss: 0.9953 - val_accuracy:
0.6426 - 65s/epoch - 2s/step
Epoch 37/150
33/33 - 65s - loss: 0.6164 - accuracy: 0.7722 - val_loss: 0.9763 - val_accuracy:
0.6350 - 65s/epoch - 2s/step
Epoch 38/150
33/33 - 67s - loss: 0.6065 - accuracy: 0.7663 - val_loss: 0.9322 - val_accuracy:
0.6616 - 67s/epoch - 2s/step
Epoch 39/150
33/33 - 67s - loss: 0.5784 - accuracy: 0.7829 - val_loss: 0.9703 - val_accuracy:
0.6350 - 67s/epoch - 2s/step
Epoch 40/150
33/33 - 68s - loss: 0.5516 - accuracy: 0.7926 - val_loss: 0.9891 - val_accuracy:
0.6540 - 68s/epoch - 2s/step
Epoch 41/150
33/33 - 67s - loss: 0.5693 - accuracy: 0.7887 - val_loss: 0.9859 - val_accuracy:
0.6350 - 67s/epoch - 2s/step
Epoch 42/150
33/33 - 66s - loss: 0.5843 - accuracy: 0.7692 - val_loss: 0.9516 - val_accuracy:
0.6540 - 66s/epoch - 2s/step
Epoch 43/150
33/33 - 65s - loss: 0.5564 - accuracy: 0.8092 - val_loss: 0.9113 - val_accuracy:
0.6502 - 65s/epoch - 2s/step
Epoch 44/150
33/33 - 65s - loss: 0.5218 - accuracy: 0.8111 - val_loss: 0.9401 - val_accuracy:
```

```
0.6654 - 65s/epoch - 2s/step
Epoch 45/150
33/33 - 64s - loss: 0.5226 - accuracy: 0.8111 - val_loss: 0.9629 - val_accuracy:
0.6464 - 64s/epoch - 2s/step
Epoch 46/150
33/33 - 65s - loss: 0.5718 - accuracy: 0.7916 - val_loss: 0.9656 - val_accuracy:
0.6616 - 65s/epoch - 2s/step
Epoch 47/150
33/33 - 64s - loss: 0.5280 - accuracy: 0.8053 - val_loss: 0.9370 - val_accuracy:
0.6654 - 64s/epoch - 2s/step
Epoch 48/150
33/33 - 65s - loss: 0.5633 - accuracy: 0.7936 - val_loss: 0.9323 - val_accuracy:
0.6350 - 65s/epoch - 2s/step
Epoch 49/150
33/33 - 64s - loss: 0.5088 - accuracy: 0.8247 - val_loss: 0.9526 - val_accuracy:
0.6426 - 64s/epoch - 2s/step
Epoch 50/150
33/33 - 65s - loss: 0.5215 - accuracy: 0.8130 - val_loss: 0.9962 - val_accuracy:
0.6312 - 65s/epoch - 2s/step
Epoch 51/150
33/33 - 64s - loss: 0.5286 - accuracy: 0.8033 - val_loss: 0.9761 - val_accuracy:
0.6464 - 64s/epoch - 2s/step
Epoch 52/150
33/33 - 65s - loss: 0.5289 - accuracy: 0.8043 - val_loss: 0.9921 - val_accuracy:
0.6540 - 65s/epoch - 2s/step
Epoch 53/150
33/33 - 64s - loss: 0.4976 - accuracy: 0.8296 - val_loss: 0.9589 - val_accuracy:
0.6578 - 64s/epoch - 2s/step
Epoch 54/150
33/33 - 65s - loss: 0.4796 - accuracy: 0.8354 - val_loss: 0.9249 - val_accuracy:
0.6578 - 65s/epoch - 2s/step
Epoch 55/150
33/33 - 64s - loss: 0.4900 - accuracy: 0.8150 - val_loss: 1.0741 - val_accuracy:
0.6464 - 64s/epoch - 2s/step
Epoch 56/150
33/33 - 66s - loss: 0.4747 - accuracy: 0.8218 - val_loss: 0.9192 - val_accuracy:
0.6692 - 66s/epoch - 2s/step
Epoch 57/150
33/33 - 64s - loss: 0.4798 - accuracy: 0.8189 - val_loss: 0.9688 - val_accuracy:
0.6730 - 64s/epoch - 2s/step
Epoch 58/150
33/33 - 66s - loss: 0.4713 - accuracy: 0.8267 - val_loss: 1.0875 - val_accuracy:
0.6616 - 66s/epoch - 2s/step
Epoch 59/150
33/33 - 65s - loss: 0.4632 - accuracy: 0.8306 - val_loss: 0.9671 - val_accuracy:
0.6616 - 65s/epoch - 2s/step
Epoch 60/150
33/33 - 65s - loss: 0.4871 - accuracy: 0.8208 - val_loss: 0.9846 - val_accuracy:
```

```
0.6692 - 65s/epoch - 2s/step
Epoch 61/150
33/33 - 65s - loss: 0.4491 - accuracy: 0.8500 - val_loss: 0.9321 - val_accuracy:
0.6730 - 65s/epoch - 2s/step
Epoch 62/150
33/33 - 65s - loss: 0.4506 - accuracy: 0.8325 - val_loss: 0.9464 - val_accuracy:
0.6654 - 65s/epoch - 2s/step
Epoch 63/150
33/33 - 66s - loss: 0.4338 - accuracy: 0.8510 - val_loss: 0.9576 - val_accuracy:
0.6654 - 66s/epoch - 2s/step
Epoch 64/150
33/33 - 66s - loss: 0.4381 - accuracy: 0.8432 - val_loss: 0.9455 - val_accuracy:
0.6692 - 66s/epoch - 2s/step
Epoch 65/150
33/33 - 66s - loss: 0.4527 - accuracy: 0.8306 - val_loss: 0.9129 - val_accuracy:
0.6920 - 66s/epoch - 2s/step
Epoch 66/150
33/33 - 66s - loss: 0.4462 - accuracy: 0.8306 - val_loss: 0.9930 - val_accuracy:
0.6730 - 66s/epoch - 2s/step
Epoch 67/150
33/33 - 66s - loss: 0.4448 - accuracy: 0.8296 - val_loss: 0.9821 - val_accuracy:
0.6692 - 66s/epoch - 2s/step
Epoch 68/150
33/33 - 68s - loss: 0.4150 - accuracy: 0.8413 - val_loss: 0.9906 - val_accuracy:
0.6502 - 68s/epoch - 2s/step
Epoch 69/150
33/33 - 68s - loss: 0.4452 - accuracy: 0.8432 - val_loss: 0.9448 - val_accuracy:
0.6730 - 68s/epoch - 2s/step
Epoch 70/150
33/33 - 66s - loss: 0.4023 - accuracy: 0.8452 - val_loss: 0.9979 - val_accuracy:
0.6654 - 66s/epoch - 2s/step
Epoch 71/150
33/33 - 65s - loss: 0.4281 - accuracy: 0.8432 - val_loss: 0.9417 - val_accuracy:
0.6768 - 65s/epoch - 2s/step
Epoch 72/150
33/33 - 67s - loss: 0.4119 - accuracy: 0.8539 - val_loss: 0.9638 - val_accuracy:
0.6616 - 67s/epoch - 2s/step
Epoch 73/150
33/33 - 66s - loss: 0.4533 - accuracy: 0.8423 - val_loss: 1.0181 - val_accuracy:
0.6464 - 66s/epoch - 2s/step
Epoch 74/150
33/33 - 66s - loss: 0.4055 - accuracy: 0.8578 - val_loss: 0.9834 - val_accuracy:
0.6616 - 66s/epoch - 2s/step
Epoch 75/150
33/33 - 65s - loss: 0.3842 - accuracy: 0.8656 - val_loss: 0.9775 - val_accuracy:
0.6882 - 65s/epoch - 2s/step
Epoch 76/150
33/33 - 66s - loss: 0.3821 - accuracy: 0.8783 - val_loss: 1.0601 - val_accuracy:
```

```
0.6578 - 66s/epoch - 2s/step
Epoch 77/150
33/33 - 64s - loss: 0.3934 - accuracy: 0.8637 - val_loss: 1.0127 - val_accuracy:
0.6768 - 64s/epoch - 2s/step
Epoch 78/150
33/33 - 66s - loss: 0.4156 - accuracy: 0.8578 - val_loss: 0.9796 - val_accuracy:
0.6882 - 66s/epoch - 2s/step
Epoch 79/150
33/33 - 64s - loss: 0.3609 - accuracy: 0.8763 - val_loss: 1.0254 - val_accuracy:
0.6692 - 64s/epoch - 2s/step
Epoch 80/150
33/33 - 65s - loss: 0.3934 - accuracy: 0.8530 - val_loss: 0.9957 - val_accuracy:
0.6730 - 65s/epoch - 2s/step
Epoch 81/150
33/33 - 65s - loss: 0.4125 - accuracy: 0.8520 - val_loss: 0.9484 - val_accuracy:
0.6768 - 65s/epoch - 2s/step
Epoch 82/150
33/33 - 65s - loss: 0.4042 - accuracy: 0.8685 - val_loss: 0.9824 - val_accuracy:
0.6882 - 65s/epoch - 2s/step
Epoch 83/150
33/33 - 65s - loss: 0.3515 - accuracy: 0.8695 - val_loss: 0.9290 - val_accuracy:
0.7034 - 65s/epoch - 2s/step
Epoch 84/150
33/33 - 66s - loss: 0.3538 - accuracy: 0.8841 - val_loss: 0.9880 - val_accuracy:
0.6882 - 66s/epoch - 2s/step
Epoch 85/150
33/33 - 65s - loss: 0.3770 - accuracy: 0.8685 - val_loss: 0.9382 - val_accuracy:
0.7148 - 65s/epoch - 2s/step
Epoch 86/150
33/33 - 64s - loss: 0.3731 - accuracy: 0.8676 - val_loss: 0.9434 - val_accuracy:
0.7262 - 64s/epoch - 2s/step
Epoch 87/150
33/33 - 65s - loss: 0.3682 - accuracy: 0.8773 - val_loss: 0.9773 - val_accuracy:
0.7224 - 65s/epoch - 2s/step
Epoch 88/150
33/33 - 65s - loss: 0.3539 - accuracy: 0.8793 - val_loss: 0.9869 - val_accuracy:
0.7034 - 65s/epoch - 2s/step
Epoch 89/150
33/33 - 64s - loss: 0.3415 - accuracy: 0.8793 - val_loss: 0.9361 - val_accuracy:
0.7034 - 64s/epoch - 2s/step
Epoch 90/150
33/33 - 65s - loss: 0.3321 - accuracy: 0.8822 - val_loss: 0.9558 - val_accuracy:
0.6768 - 65s/epoch - 2s/step
Epoch 91/150
33/33 - 65s - loss: 0.3385 - accuracy: 0.8744 - val_loss: 0.9208 - val_accuracy:
0.6996 - 65s/epoch - 2s/step
Epoch 92/150
33/33 - 65s - loss: 0.3506 - accuracy: 0.8705 - val_loss: 0.9813 - val_accuracy:
```

```
0.6920 - 65s/epoch - 2s/step
Epoch 93/150
33/33 - 65s - loss: 0.3493 - accuracy: 0.8754 - val_loss: 0.9154 - val_accuracy:
0.7110 - 65s/epoch - 2s/step
Epoch 94/150
33/33 - 65s - loss: 0.3477 - accuracy: 0.8637 - val_loss: 0.9179 - val_accuracy:
0.7262 - 65s/epoch - 2s/step
Epoch 95/150
33/33 - 64s - loss: 0.3220 - accuracy: 0.8939 - val_loss: 0.9812 - val_accuracy:
0.7224 - 64s/epoch - 2s/step
Epoch 96/150
33/33 - 65s - loss: 0.3280 - accuracy: 0.8880 - val_loss: 0.9232 - val_accuracy:
0.6844 - 65s/epoch - 2s/step
Epoch 97/150
33/33 - 66s - loss: 0.3314 - accuracy: 0.8861 - val_loss: 0.9230 - val_accuracy:
0.6958 - 66s/epoch - 2s/step
Epoch 98/150
33/33 - 67s - loss: 0.3108 - accuracy: 0.8851 - val_loss: 0.9668 - val_accuracy:
0.7034 - 67s/epoch - 2s/step
Epoch 99/150
33/33 - 66s - loss: 0.3097 - accuracy: 0.8939 - val_loss: 0.9604 - val_accuracy:
0.7110 - 66s/epoch - 2s/step
Epoch 100/150
33/33 - 66s - loss: 0.3238 - accuracy: 0.8900 - val_loss: 1.0117 - val_accuracy:
0.6958 - 66s/epoch - 2s/step
Epoch 101/150
33/33 - 65s - loss: 0.3214 - accuracy: 0.8978 - val_loss: 1.0648 - val_accuracy:
0.6730 - 65s/epoch - 2s/step
Epoch 102/150
33/33 - 67s - loss: 0.2906 - accuracy: 0.8929 - val_loss: 1.0851 - val_accuracy:
0.6920 - 67s/epoch - 2s/step
Epoch 103/150
33/33 - 65s - loss: 0.3251 - accuracy: 0.8870 - val_loss: 0.9742 - val_accuracy:
0.7148 - 65s/epoch - 2s/step
Epoch 104/150
33/33 - 67s - loss: 0.3248 - accuracy: 0.8802 - val_loss: 1.0190 - val_accuracy:
0.6806 - 67s/epoch - 2s/step
Epoch 105/150
33/33 - 64s - loss: 0.2840 - accuracy: 0.8968 - val_loss: 1.0347 - val_accuracy:
0.6768 - 64s/epoch - 2s/step
Epoch 106/150
33/33 - 65s - loss: 0.3220 - accuracy: 0.8870 - val_loss: 1.0098 - val_accuracy:
0.6996 - 65s/epoch - 2s/step
Epoch 107/150
33/33 - 64s - loss: 0.3084 - accuracy: 0.8997 - val_loss: 1.0203 - val_accuracy:
0.6730 - 64s/epoch - 2s/step
Epoch 108/150
33/33 - 64s - loss: 0.3033 - accuracy: 0.8870 - val_loss: 0.9997 - val_accuracy:
```

```
0.6654 - 64s/epoch - 2s/step
Epoch 109/150
33/33 - 65s - loss: 0.2966 - accuracy: 0.9046 - val_loss: 1.0084 - val_accuracy:
0.6958 - 65s/epoch - 2s/step
Epoch 110/150
33/33 - 67s - loss: 0.2843 - accuracy: 0.8997 - val_loss: 1.0608 - val_accuracy:
0.6844 - 67s/epoch - 2s/step
Epoch 111/150
33/33 - 65s - loss: 0.2942 - accuracy: 0.9065 - val_loss: 1.0373 - val_accuracy:
0.6882 - 65s/epoch - 2s/step
Epoch 112/150
33/33 - 65s - loss: 0.3190 - accuracy: 0.8861 - val_loss: 1.1053 - val_accuracy:
0.6768 - 65s/epoch - 2s/step
Epoch 113/150
33/33 - 66s - loss: 0.2723 - accuracy: 0.8997 - val_loss: 1.0884 - val_accuracy:
0.6806 - 66s/epoch - 2s/step
Epoch 114/150
33/33 - 66s - loss: 0.3007 - accuracy: 0.8968 - val_loss: 1.0920 - val_accuracy:
0.6844 - 66s/epoch - 2s/step
Epoch 115/150
33/33 - 66s - loss: 0.2701 - accuracy: 0.9153 - val_loss: 1.1078 - val_accuracy:
0.6920 - 66s/epoch - 2s/step
Epoch 116/150
33/33 - 68s - loss: 0.2550 - accuracy: 0.9192 - val_loss: 1.0668 - val_accuracy:
0.6844 - 68s/epoch - 2s/step
Epoch 117/150
33/33 - 67s - loss: 0.2882 - accuracy: 0.9017 - val_loss: 1.0314 - val_accuracy:
0.6920 - 67s/epoch - 2s/step
Epoch 118/150
33/33 - 67s - loss: 0.2941 - accuracy: 0.9017 - val_loss: 1.0369 - val_accuracy:
0.6844 - 67s/epoch - 2s/step
Epoch 119/150
33/33 - 68s - loss: 0.2980 - accuracy: 0.9017 - val_loss: 1.0482 - val_accuracy:
0.6730 - 68s/epoch - 2s/step
Epoch 120/150
33/33 - 67s - loss: 0.2420 - accuracy: 0.9192 - val_loss: 1.0646 - val_accuracy:
0.6844 - 67s/epoch - 2s/step
Epoch 121/150
33/33 - 66s - loss: 0.2931 - accuracy: 0.8948 - val_loss: 1.0163 - val_accuracy:
0.6958 - 66s/epoch - 2s/step
Epoch 122/150
33/33 - 65s - loss: 0.2495 - accuracy: 0.9182 - val_loss: 1.0198 - val_accuracy:
0.6844 - 65s/epoch - 2s/step
Epoch 123/150
33/33 - 66s - loss: 0.2715 - accuracy: 0.8987 - val_loss: 1.0456 - val_accuracy:
0.6958 - 66s/epoch - 2s/step
Epoch 124/150
33/33 - 66s - loss: 0.2569 - accuracy: 0.9172 - val_loss: 1.1174 - val_accuracy:
```

```
0.6844 - 66s/epoch - 2s/step
Epoch 125/150
33/33 - 65s - loss: 0.2497 - accuracy: 0.9104 - val_loss: 1.0495 - val_accuracy:
0.7072 - 65s/epoch - 2s/step
Epoch 126/150
33/33 - 66s - loss: 0.2845 - accuracy: 0.9056 - val_loss: 1.0720 - val_accuracy:
0.7034 - 66s/epoch - 2s/step
Epoch 127/150
33/33 - 66s - loss: 0.2632 - accuracy: 0.9143 - val_loss: 1.0895 - val_accuracy:
0.7110 - 66s/epoch - 2s/step
Epoch 128/150
33/33 - 66s - loss: 0.2712 - accuracy: 0.9026 - val_loss: 0.9912 - val_accuracy:
0.6920 - 66s/epoch - 2s/step
Epoch 129/150
33/33 - 65s - loss: 0.2372 - accuracy: 0.9211 - val_loss: 0.9947 - val_accuracy:
0.6996 - 65s/epoch - 2s/step
Epoch 130/150
33/33 - 66s - loss: 0.2828 - accuracy: 0.9172 - val_loss: 1.0127 - val_accuracy:
0.7072 - 66s/epoch - 2s/step
Epoch 131/150
33/33 - 65s - loss: 0.2267 - accuracy: 0.9289 - val_loss: 0.9790 - val_accuracy:
0.7034 - 65s/epoch - 2s/step
Epoch 132/150
33/33 - 65s - loss: 0.2394 - accuracy: 0.9211 - val_loss: 1.1154 - val_accuracy:
0.6882 - 65s/epoch - 2s/step
Epoch 133/150
33/33 - 65s - loss: 0.2377 - accuracy: 0.9202 - val_loss: 0.9881 - val_accuracy:
0.6958 - 65s/epoch - 2s/step
Epoch 134/150
33/33 - 65s - loss: 0.2288 - accuracy: 0.9202 - val_loss: 1.0871 - val_accuracy:
0.7072 - 65s/epoch - 2s/step
Epoch 135/150
33/33 - 65s - loss: 0.2218 - accuracy: 0.9270 - val_loss: 1.0911 - val_accuracy:
0.6958 - 65s/epoch - 2s/step
Epoch 136/150
33/33 - 66s - loss: 0.2504 - accuracy: 0.9124 - val_loss: 1.1095 - val_accuracy:
0.6996 - 66s/epoch - 2s/step
Epoch 137/150
33/33 - 65s - loss: 0.2544 - accuracy: 0.9124 - val_loss: 1.0782 - val_accuracy:
0.6920 - 65s/epoch - 2s/step
Epoch 138/150
33/33 - 66s - loss: 0.2322 - accuracy: 0.9309 - val_loss: 1.0731 - val_accuracy:
0.7072 - 66s/epoch - 2s/step
Epoch 139/150
33/33 - 65s - loss: 0.2196 - accuracy: 0.9338 - val_loss: 1.0610 - val_accuracy:
0.6920 - 65s/epoch - 2s/step
Epoch 140/150
33/33 - 66s - loss: 0.2415 - accuracy: 0.9172 - val_loss: 1.0806 - val_accuracy:
```

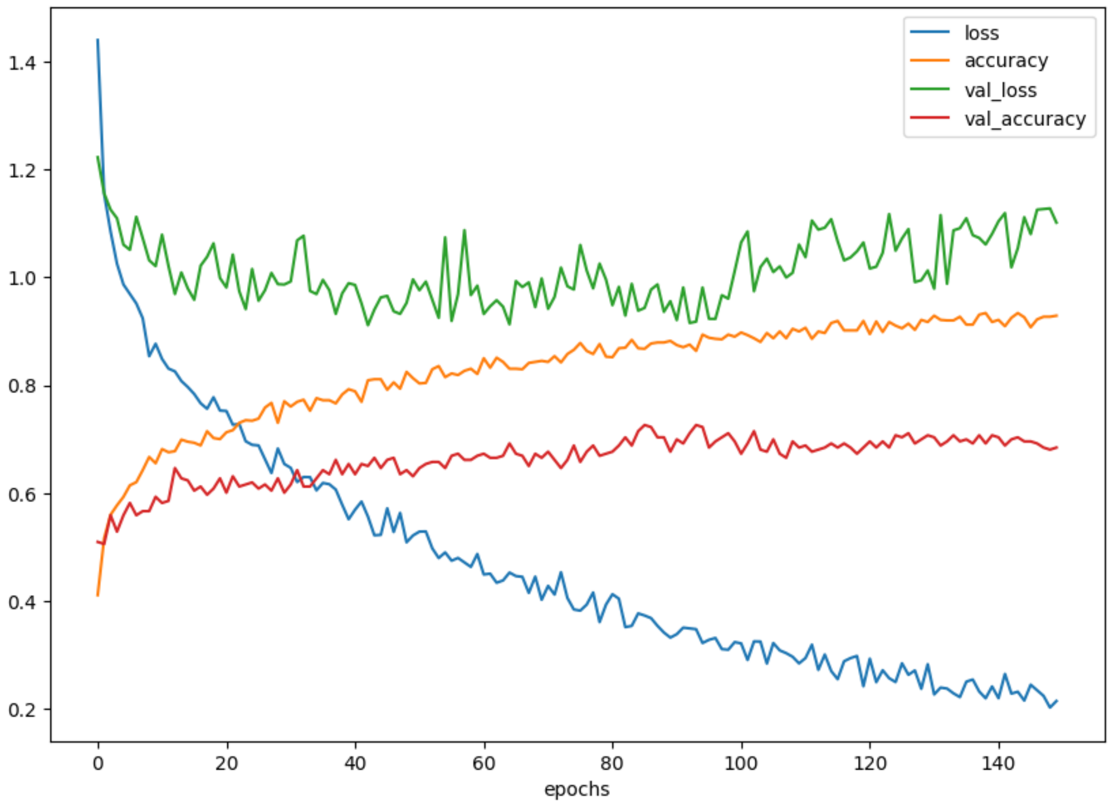
```
0.7072 - 66s/epoch - 2s/step
Epoch 141/150
33/33 - 65s - loss: 0.2197 - accuracy: 0.9211 - val_loss: 1.1040 - val_accuracy:
0.7034 - 65s/epoch - 2s/step
Epoch 142/150
33/33 - 65s - loss: 0.2647 - accuracy: 0.9094 - val_loss: 1.1191 - val_accuracy:
0.6882 - 65s/epoch - 2s/step
Epoch 143/150
33/33 - 66s - loss: 0.2283 - accuracy: 0.9250 - val_loss: 1.0186 - val_accuracy:
0.6996 - 66s/epoch - 2s/step
Epoch 144/150
33/33 - 65s - loss: 0.2318 - accuracy: 0.9338 - val_loss: 1.0540 - val_accuracy:
0.7034 - 65s/epoch - 2s/step
Epoch 145/150
33/33 - 65s - loss: 0.2155 - accuracy: 0.9260 - val_loss: 1.1115 - val_accuracy:
0.6958 - 65s/epoch - 2s/step
Epoch 146/150
33/33 - 66s - loss: 0.2448 - accuracy: 0.9075 - val_loss: 1.0802 - val_accuracy:
0.6958 - 66s/epoch - 2s/step
Epoch 147/150
33/33 - 65s - loss: 0.2345 - accuracy: 0.9221 - val_loss: 1.1255 - val_accuracy:
0.6920 - 65s/epoch - 2s/step
Epoch 148/150
33/33 - 65s - loss: 0.2241 - accuracy: 0.9270 - val_loss: 1.1265 - val_accuracy:
0.6844 - 65s/epoch - 2s/step
Epoch 149/150
33/33 - 65s - loss: 0.2024 - accuracy: 0.9270 - val_loss: 1.1278 - val_accuracy:
0.6806 - 65s/epoch - 2s/step
Epoch 150/150
33/33 - 66s - loss: 0.2145 - accuracy: 0.9289 - val_loss: 1.1015 - val_accuracy:
0.6844 - 66s/epoch - 2s/step
```

[44]:

```python
import pandas as pd
```

[45]:

```python
pd.DataFrame(history.history).plot(figsize = (10, 7), xlabel = "epochs");
```



[46]:

```
# #plot the learning rate versus the loss
# lrs = 1e-4 * (10 ** (tf.range(300)/20))
# plt.figure(figsize = (10,7))
# plt.semilogx(lrs, history.history["loss"])
# plt.xlabel("learning rate")
# plt.ylabel("loss")
# plt.title("learning rate vs loss")
```

[47]:

```
%matplotlib inline

import matplotlib.image as mpimg
import matplotlib.pyplot as plt
```

[48]:

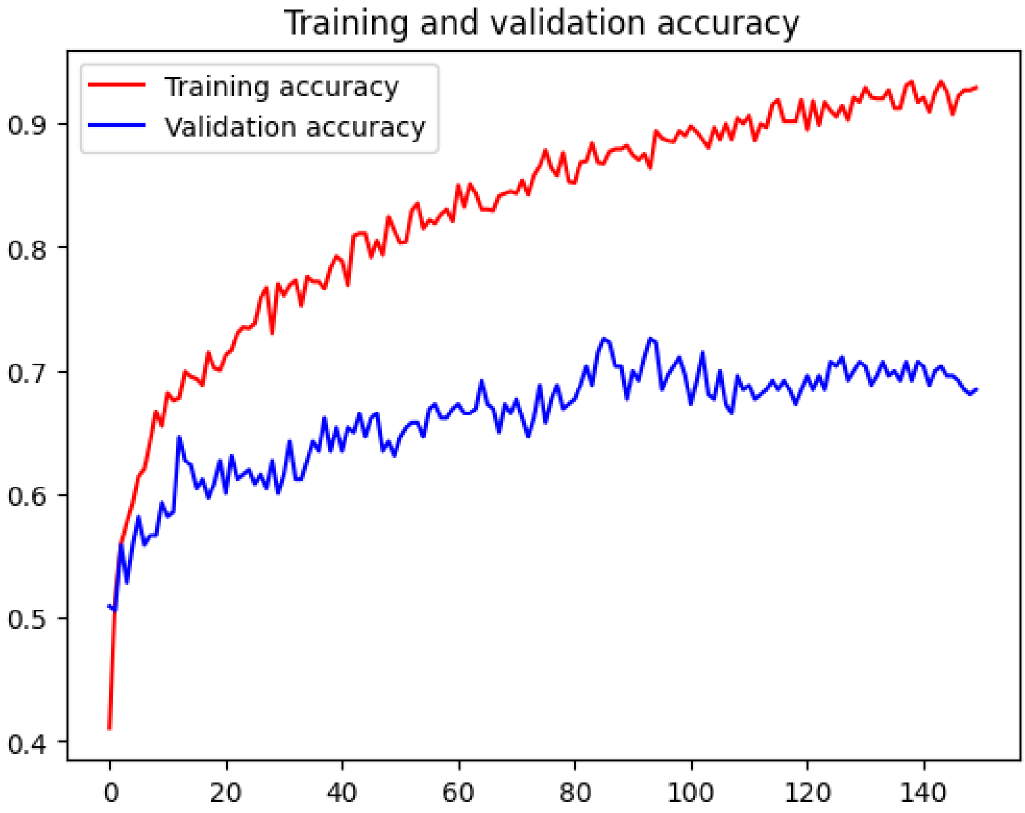
```
# evaluating accuracy and loss for the model
```

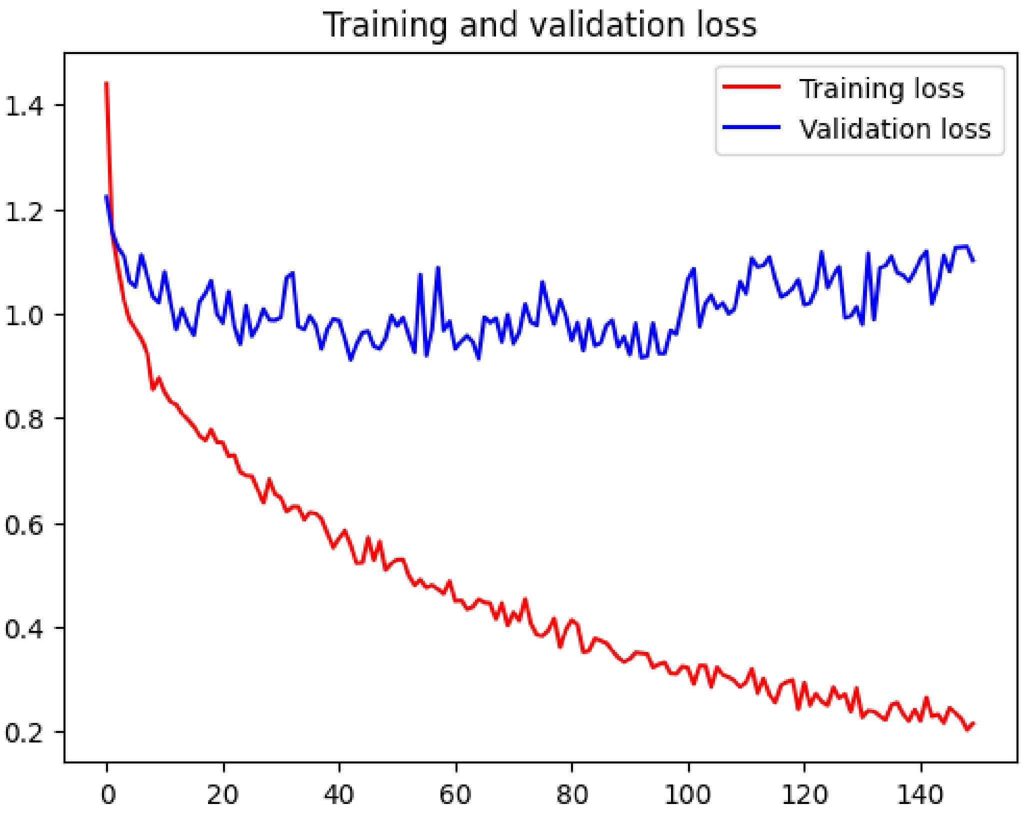
[49]:

```
#-----------------------------------------------------------
# Retrieve a list of list results on training and test data
# sets for each training epoch
#-----------------------------------------------------------
acc      = history.history[     'accuracy' ]
val_acc  = history.history[ 'val_accuracy' ]
```

```
loss     = history.history[    'loss' ]
val_loss = history.history['val_loss' ]

epochs   = range(len(acc)) # Get number of epochs

#------------------------------------------------
# Plot training and validation accuracy per epoch
#------------------------------------------------
plt.plot(epochs, acc, 'r', label='Training accuracy')
plt.plot(epochs, val_acc, 'b', label='Validation accuracy')
plt.title ('Training and validation accuracy')
plt.legend(loc=0)
plt.figure()

#------------------------------------------------
# Plot training and validation loss per epoch
#------------------------------------------------
plt.plot  ( epochs,     loss , 'r', label='Training loss')
plt.plot  ( epochs, val_loss , 'b', label='Validation loss')
plt.title ('Training and validation loss'   )
plt.legend(loc=0)
plt.figure()
```

```
[49]: <Figure size 640x480 with 0 Axes>
```


Training and validation accuracy
Training accuracy
Validation accuracy
0.9
0.8
0.7
0.6
0.5
0.4
0
20
40
60
80
100
120
140



```
<Figure size 640x480 with 0 Axes>
```

## 3 menyimpan model TAPI BELUM YANG VERSI QUANTIZED

[35]:
```
from keras.models import load_model

model.save('/home/jupyter/content/kerasFormat_model/model_vggFace_11Juni.h5')
```

[36]:
```
from keras.models import load_model

model.save('/home/jupyter/content/saved Model')
```

```
WARNING:absl:Found untraced functions such as _jit_compiled_convolution_op,
_jit_compiled_convolution_op, _jit_compiled_convolution_op,
_jit_compiled_convolution_op, _jit_compiled_convolution_op while saving (showing
5 of 16). These functions will not be directly callable after loading.

INFO:tensorflow:Assets written to: /home/jupyter/content/saved Model5/assets

INFO:tensorflow:Assets written to: /home/jupyter/content/saved Model5/assets
```

# 4 Convert to Tensorflow LITE

[ ]:
```
try:
    %tensorflow_version 2.x
except:
    pass
```

[ ]:
```
import pathlib

print('\u2022 Using TensorFlow Version:', tf.__version__)
```

[ ]:
```

```

[ ]:
```
#!mkdir saved_TFLITE_model
```

generate SavedModel

[ ]:
```
export_dir = "home/jupyter/content/saved_TFLITE_model"
tf.saved_model.save(model, export_dir)
```

[ ]:
```
model = tf.saved_model.load(export_dir)
```

Convert the SavedModel to JSON

[ ]:
```
#!pip install --user tensorflowjs
```

[ ]:
```
# import tensorflowjs
# import json
```

[ ]:
```
# # Load the SavedModel
# model = tf.saved_model.load('home/jupyter/content/saved Model')
```

[ ]:
```
import tensorflow as tf
import json

# Load the SavedModel
model = tf.saved_model.load(export_dir)

# Convert tensor shapes to lists
def convert_shape(shape):
    return [dim for dim in shape.as_list()]

# Create a dictionary to store the JSON model
json_model = {}

# Get information about inputs
input_signatures = model.signatures['serving_default'].
 ↪structured_input_signature[1]
json_model['inputs'] = []
```

```python
for tensor_name, tensor_info in input_signatures.items():
    input_info = {
        'name': tensor_name,
        'dtype': str(tensor_info.dtype),
        'shape': convert_shape(tensor_info.shape),
    }
    json_model['inputs'].append(input_info)

# Get information about outputs
output_signatures = model.signatures['serving_default'].structured_outputs
json_model['outputs'] = []
for tensor_name, tensor_info in output_signatures.items():
    output_info = {
        'name': tensor_name,
        'dtype': str(tensor_info.dtype),
        'shape': convert_shape(tensor_info.shape),
    }
    json_model['outputs'].append(output_info)

# Save the JSON model to a file
with open('model.json', 'w') as f:
    json.dump(json_model, f)
```

Convert the SavedModel to TFLite

[ ]:
```python
# # Convert the model.
# converter = tf.lite.TFLiteConverter.from_saved_model(export_dir)

# # Set the optimization flags for quantization
# converter.optimizations = [tf.lite.Optimize.DEFAULT] # INTINYA DISINI UNTUK␣
 ↪MELAKUKAN QUANTIZED

# # Specify the input and output tensors (if necessary)
# # converter.target_spec.supported_ops = [tf.lite.OpsSet.TFLITE_BUILTINS_INT8]
# # converter.target_spec.supported_types = [tf.int8]

# tflite_model = converter.convert()
```

[ ]:
```python
# tflite_model_file = pathlib.Path('home/jupyter/content/')
# tflite_model_file.write_bytes(tflite_model)
```

## 5 labels

[ ]:
```python
class_names = ['combination', 'dry', 'nonface' ,'normal', 'oily', 'sensitive']
```

create a txt file to save the 5 labels

[ ]:

```
with open('faces_labels.txt', 'w') as f:
    f.write('\n'.join(class_names))
```

[ ]:

```
import shutil
shutil.copy("/content/faces_labels.txt","/content/drive/MyDrive/saved model")
```

[ ]:

```
from google.colab import drive
drive.mount('/content/drive')
```

[ ]:

```
import os
file_path = "/home/jupyter/content/kerasFormat_model/model_vggFace_10Juni.h5" ␣
 ↪# Replace with the correct file path
file_size = os.path.getsize(file_path)
print(file_size)
```

[ ]:

```
!wget -O /home/jupyter/content/kerasFormat_model/model_vggFace_10Juni.h5 https:/
 ↪/58948e97954de7e0-dot-asia-southeast2.notebooks.googleusercontent.com/files/
 ↪content/kerasFormat_model/model_vggFace_10Juni.h5?
 ↪_xsrf=2%7C1f091a48%7C9c97e6fd85cef3a278dc9a8e0d4188fa%7C1686143743
```

## 6 Perform inference

[50]:

```
# inference dengan model machine learning format keras .h5

def preprocess_image(image):
    # Normalize pixel values to the range [0, 1]
    image = image / 255.0

    # Perform any other preprocessing steps such as resizing, cropping, etc.
    # ...

    return image

import tensorflow as tf
from tensorflow.keras.models import load_model

# Load the model
model = load_model('/home/jupyter/content/kerasFormat_model/
 ↪model_vggFace_11Juni.h5')

# Load and preprocess the image
image = tf.keras.preprocessing.image.load_img('/home/jupyter/content/testFace/
 ↪oily/oily_testIMG.jpg', target_size=(224,224))
image = tf.keras.preprocessing.image.img_to_array(image)
image = preprocess_image(image)  # Preprocess the image as per your model's␣
 ↪requirements
```

```
# Add a batch dimension to the image
image = tf.expand_dims(image, axis=0)

# Perform inference
predictions = model.predict(image)

# Process the predictions as per your requirements
```

```
1/1 [==============================] - 1s 1s/step
```

[51]:

```
# Perform inference
predictions = model(image)

# Access the predicted class or values
predicted_classes = tf.argmax(predictions, axis=1)
print("Predicted classes:", predicted_classes)

# Or, access specific elements of the predictions
# prediction_value = predictions[0][0]  # Example for a specific element

# Process the predictions further as per your requirements
```

```
Predicted classes: tf.Tensor([0], shape=(1,), dtype=int64)
```

[52]:

```
class_labels = {0: 'Oily', 1: 'Normal',2: 'Sensitive' , 3: 'Dry', 4:'nonface',␣
  ↪5:"Combination" }  # Example mapping of class indices to labels

predicted_classes = tf.argmax(predictions, axis=1)

for predicted_class in predicted_classes:
    predicted_label = class_labels[predicted_class.numpy()]
    print("Predicted label:", predicted_label)
```

```
Predicted label: Oily
```

[ ]: