Project – Ocular Disease Image Classification (work in progress)

October 16, 2020

[1]:
```
import numpy as np
import math
import pandas as pd
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Activation, Dense, Flatten,␣
 ↪BatchNormalization, Conv2D, MaxPool2D, Dropout
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.metrics import categorical_crossentropy
from tensorflow.keras.preprocessing.image import ImageDataGenerator,␣
 ↪img_to_array, load_img
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder, LabelEncoder
import itertools
import os
import shutil
import random
import glob
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud, STOPWORDS
import warnings
warnings.simplefilter(action = 'ignore', category = FutureWarning)
import re
from PIL import Image

%matplotlib inline
```

[2]:
```
eye = pd.read_excel('/Users/armaanvalvi/Documents/project-data/archive/ODIR-5K/
 ↪data.xlsx')
```

[3]:
```
eye.columns = ['id', 'age', 'sex', 'left', 'right', 'l-diagnosis',␣
 ↪'r-diagnosis', 'normal', 'diabetes', 'glaucoma',
               'cataract', 'amd', 'hypertension', 'myopia', 'other']
```

[4]:
```
eye2 = pd.read_csv('/Users/armaanvalvi/Documents/project-data/archive/full_df.
↪csv')
eye2.head(2)
```

[4]:
```
   ID  Patient Age Patient Sex Left-Fundus Right-Fundus  \
0   0           69      Female  0_left.jpg  0_right.jpg
1   1           57        Male  1_left.jpg  1_right.jpg

  Left-Diagnostic Keywords Right-Diagnostic Keywords  N  D  G  C  A  H  M  O  \
0                 cataract             normal fundus  0  0  0  1  0  0  0  0
1            normal fundus             normal fundus  1  0  0  0  0  0  0  0

                                           filepath labels  \
0  ../input/ocular-disease-recognition-odir5k/ODI…  ['N']
1  ../input/ocular-disease-recognition-odir5k/ODI…  ['N']

                     target     filename
0  [1, 0, 0, 0, 0, 0, 0, 0]  0_right.jpg
1  [1, 0, 0, 0, 0, 0, 0, 0]  1_right.jpg
```

[5]:
```
len(eye2.filename.unique())
```

[5]:
```
6392
```

[6]:
```
eye2.columns = ['id', 'age', 'sex', 'left', 'right', 'l-diagnosis',␣
↪'r-diagnosis', 'normal', 'diabetes', 'glaucoma',
             'cataract', 'amd', 'hypertension', 'myopia', 'other', 'filepath',␣
↪'labels', 'target', 'file_name']
```

[7]:
```
eye2.drop(['filepath'], axis = 1, inplace = True)
```

[8]:
```
eye2.loc[eye2['sex'] == 'Male', 'sex'] = 'M'
eye2.loc[eye2['sex'] == 'Female', 'sex'] = 'F'
# eye2.rename(columns = {'hypertension':'h-tension'})
eye2.head(2)
```

[8]:
```
   id  age sex        left        right    l-diagnosis    r-diagnosis  normal  \
0   0   69   F  0_left.jpg  0_right.jpg       cataract  normal fundus       0
1   1   57   M  1_left.jpg  1_right.jpg  normal fundus  normal fundus       1

   diabetes  glaucoma  cataract  amd  hypertension  myopia  other labels  \
0         0         0         1    0             0       0      0  ['N']
1         0         0         0    0             0       0      0  ['N']

                     target    file_name
0  [1, 0, 0, 0, 0, 0, 0, 0]  0_right.jpg
1  [1, 0, 0, 0, 0, 0, 0, 0]  1_right.jpg
```

[9]:
```
list_columns = eye2.columns.tolist()
list_columns
```

[9]:
```
['id',
 'age',
 'sex',
 'left',
 'right',
 'l-diagnosis',
 'r-diagnosis',
 'normal',
 'diabetes',
 'glaucoma',
 'cataract',
 'amd',
 'hypertension',
 'myopia',
 'other',
 'labels',
 'target',
 'file_name']
```

[10]:
```
eye_final = eye2[[list_columns[-1]] + list_columns[0:5] + list_columns[15:17] +␣
↪list_columns[5:15]]
eye_final.head(2)
```

[10]:

	file_name	id	age	sex	left	right	labels	target	l-diagnosis	r-diagnosis	normal	diabetes	glaucoma	cataract	amd	hypertension	myopia	other
0	0_right.jpg	0	69	F	0_left.jpg	0_right.jpg	['N']	[1, 0, 0, 0, 0, 0, 0, 0]	cataract	normal fundus	0	0	0	1	0	0	0	0
1	1_right.jpg	1	57	M	1_left.jpg	1_right.jpg	['N']	[1, 0, 0, 0, 0, 0, 0, 0]	normal fundus	normal fundus	1	0	0	0	0	0	0	0

[11]:
```
eye_final.loc[eye_final['labels'] == "['N']", 'labels'] = 'N'
eye_final.loc[eye_final['labels'] == "['D']", 'labels'] = 'D'
eye_final.loc[eye_final['labels'] == "['G']", 'labels'] = 'G'
eye_final.loc[eye_final['labels'] == "['C']", 'labels'] = 'C'
eye_final.loc[eye_final['labels'] == "['A']", 'labels'] = 'A'
eye_final.loc[eye_final['labels'] == "['H']", 'labels'] = 'H'
eye_final.loc[eye_final['labels'] == "['M']", 'labels'] = 'M'
eye_final.loc[eye_final['labels'] == "['O']", 'labels'] = 'O'
```

[12]:
```
eye_final.head(2)
```

[12]:
```
     file_name  id  age sex        left        right labels  \
0  0_right.jpg   0   69   F  0_left.jpg  0_right.jpg      N
1  1_right.jpg   1   57   M  1_left.jpg  1_right.jpg      N

                     target    l-diagnosis    r-diagnosis  normal  diabetes  \
0  [1, 0, 0, 0, 0, 0, 0, 0]       cataract  normal fundus       0         0
1  [1, 0, 0, 0, 0, 0, 0, 0]  normal fundus  normal fundus       1         0

   glaucoma  cataract  amd  hypertension  myopia  other
0         0         1    0             0       0      0
1         0         0    0             0       0      0
```

[13]:
```
files = glob.glob('/Users/armaanvalvi/Documents/project-data/archive/images/*.
↪jpg')
data = []
for file in files:
#     img = Image.open(file)
    data.append(file)

# img = Image.open(data[0])
# plt.imshow(img)
```

[14]:
```
data_df = pd.DataFrame({"file_path":data})
```

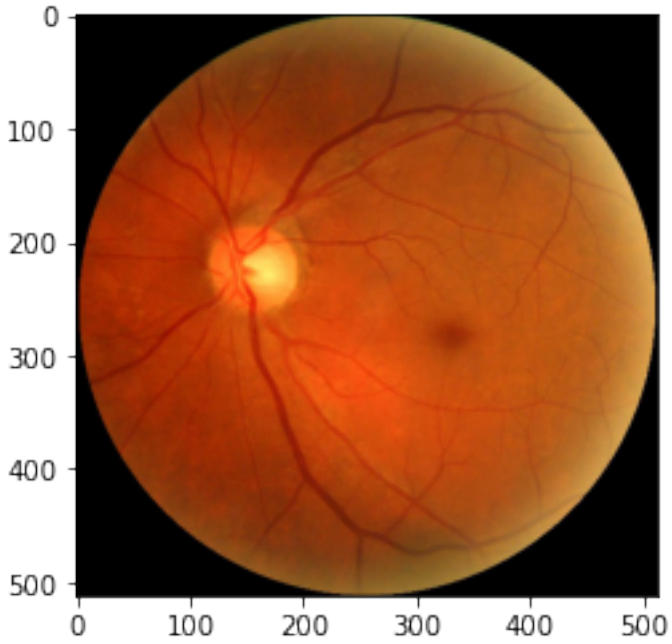
[15]:
```
eye_final = pd.concat([eye_final, data_df], axis = 1)
```

[16]:
```
eye_final.head(2)
```

[16]:
```
     file_name  id  age sex        left        right labels  \
0  0_right.jpg   0   69   F  0_left.jpg  0_right.jpg      N
1  1_right.jpg   1   57   M  1_left.jpg  1_right.jpg      N

                     target    l-diagnosis    r-diagnosis  normal  diabetes  \
0  [1, 0, 0, 0, 0, 0, 0, 0]       cataract  normal fundus       0         0
1  [1, 0, 0, 0, 0, 0, 0, 0]  normal fundus  normal fundus       1         0

   glaucoma  cataract  amd  hypertension  myopia  other  \
0         0         1    0             0       0      0
1         0         0    0             0       0      0

                                           file_path
0  /Users/armaanvalvi/Documents/project-data/arch…
1  /Users/armaanvalvi/Documents/project-data/arch…
```

[17]:
```
pilot_img = Image.open(eye_final['file_path'][0])
plt.imshow(pilot_img)

pilot_img.size
```

[17]: (512, 512)



[18]:
```
eye_final.groupby('labels').sum()
```

[18]:

labels	id	age	normal	diabetes	glaucoma	cataract	amd	hypertension	myopia	other
A	269524	16282	0	11	17	0	266	4	3	16
C	552002	19562	0	42	5	293	0	0	0	31
D	4993483	89565	0	1608	18	32	18	60	19	304
G	383847	17898	0	34	284	1	2			
H	192546	7302	0	29	6	4	4			
M	317821	12734	0	12	0	0	0			
N	7320912	164283	2101	252	42	51	22			
O	487061	42202	0	135	25	21	7			

```
G                  9      11     46
H                128       0     16
M                  0     232     41
N                  0      26    429
O                  2      15    705
```

[19]:
```
eye_final.describe()
```

[19]:
```
                id          age       normal     diabetes     glaucoma  \
count  6392.000000  6392.000000  6392.000000  6392.000000  6392.000000
mean   2271.150814    57.857947     0.328692     0.332134     0.062109
std    1417.559018    11.727737     0.469775     0.471016     0.241372
min       0.000000     1.000000     0.000000     0.000000     0.000000
25%     920.750000    51.000000     0.000000     0.000000     0.000000
50%    2419.500000    59.000000     0.000000     0.000000     0.000000
75%    3294.000000    66.000000     1.000000     1.000000     0.000000
max    4784.000000    91.000000     1.000000     1.000000     1.000000

           cataract          amd  hypertension       myopia        other
count  6392.000000  6392.000000   6392.000000  6392.000000  6392.000000
mean      0.062891     0.049906      0.031758     0.047872     0.248436
std       0.242786     0.217768      0.175370     0.213513     0.432139
min       0.000000     0.000000      0.000000     0.000000     0.000000
25%       0.000000     0.000000      0.000000     0.000000     0.000000
50%       0.000000     0.000000      0.000000     0.000000     0.000000
75%       0.000000     0.000000      0.000000     0.000000     0.000000
max       1.000000     1.000000      1.000000     1.000000     1.000000
```

[20]:
```
eye_final.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6392 entries, 0 to 6391
Data columns (total 19 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   file_name     6392 non-null   object
 1   id            6392 non-null   int64
 2   age           6392 non-null   int64
 3   sex           6392 non-null   object
 4   left          6392 non-null   object
 5   right         6392 non-null   object
 6   labels        6392 non-null   object
 7   target        6392 non-null   object
 8   l-diagnosis   6392 non-null   object
 9   r-diagnosis   6392 non-null   object
 10  normal        6392 non-null   int64
 11  diabetes      6392 non-null   int64
```

```
 12  glaucoma      6392 non-null   int64
 13  cataract      6392 non-null   int64
 14  amd           6392 non-null   int64
 15  hypertension  6392 non-null   int64
 16  myopia        6392 non-null   int64
 17  other         6392 non-null   int64
 18  file_path     6392 non-null   object
dtypes: int64(10), object(9)
memory usage: 948.9+ KB
```

[21]:
```
eye_final['labels'].value_counts()
```

[21]:
```
N    2873
D    1608
O     708
C     293
G     284
A     266
M     232
H     128
Name: labels, dtype: int64
```

[22]:
```
train_data, test_data = train_test_split(eye_final, test_size = 0.18,␣
↪random_state = 1234)
```

[23]:
```
train_data['labels'].value_counts(), test_data['labels'].value_counts()
```

[23]:
```
(N    2351
 D    1323
 O     598
 C     235
 G     229
 A     212
 M     189
 H     104
 Name: labels, dtype: int64,
 N    522
 D    285
 O    110
 C     58
 G     55
 A     54
 M     43
 H     24
 Name: labels, dtype: int64)
```

[24]:
```
train_data.loc[train_data['sex'] == 'M', 'sex'] = '0'
train_data.loc[train_data['sex'] == 'F', 'sex'] = '1'
train_data.drop(['right'], axis = 1, inplace = True)
```

```
/Users/armaanvalvi/opt/anaconda3/envs/main/lib/python3.7/site-
packages/pandas/core/indexing.py:1765: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  isetter(loc, value)
/Users/armaanvalvi/opt/anaconda3/envs/main/lib/python3.7/site-
packages/pandas/core/frame.py:4164: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  errors=errors,
```

[25]:
```
train_data.loc[train_data['file_name'].str.contains('left', regex = True),␣
↪'left'] = '1'
train_data.loc[train_data['file_name'].str.contains('right', regex = True),␣
↪'left'] = '0'
train_data.head(2)
```

[25]:

	file_name	id	age	sex	left	labels	target	\
1941	2799_right.jpg	2799	53	0	0	N	[1, 0, 0, 0, 0, 0, 0, 0]	
6391	4784_left.jpg	4784	58	0	1	H	[0, 0, 0, 0, 0, 1, 0, 0]	

	l-diagnosis	\
1941	normal fundus	
6391	hypertensive retinopathy age-related macular d…	

	r-diagnosis	normal	diabetes	\
1941	normal fundus	1	0	
6391	hypertensive retinopathy age-related macular d…	0	0	

	glaucoma	cataract	amd	hypertension	myopia	other	\
1941	0	0	0	0	0	0	
6391	0	0	1	1	0	0	

	file_path
1941	/Users/armaanvalvi/Documents/project-data/arch…
6391	/Users/armaanvalvi/Documents/project-data/arch…

[26]:
```
train_data.columns
```

[26]:
```
Index(['file_name', 'id', 'age', 'sex', 'left', 'labels', 'target',
       'l-diagnosis', 'r-diagnosis', 'normal', 'diabetes', 'glaucoma',
       'cataract', 'amd', 'hypertension', 'myopia', 'other', 'file_path'],
      dtype='object')
```

[27]:
```
train_final = pd.DataFrame(train_data, columns =['sex', 'left'])
one_hot_columns = train_final.columns.tolist()
train_final_enc = train_final.apply(LabelEncoder().fit_transform)
train_diagnosis = pd.DataFrame(train_data, columns = ['age', 'normal',
                                                      'diabetes', 'glaucoma',
                                                      'cataract', 'amd',
                                                      'hypertension', 'myopia',
                                                      'other'
                                                     ])
train_final_enc = pd.concat([train_final_enc, train_diagnosis], axis = 1)
```

[28]:
```
train_final_enc
X_train = train_final_enc
y_train = train_data['target']
```

[29]:
```
len(test_data)
```

[29]:
```
1151
```

0.1 Creating more pictures to create a more balanced image dataset

[30]:
```
"""create lists of all the file_names for each of the different conditions and␣
 ↪keep them ready"""


n_names = list(eye_final['file_name'][eye_final['labels'] == 'N'])
d_names = list(eye_final['file_name'][eye_final['labels'] == 'D'])
o_names = list(eye_final['file_name'][eye_final['labels'] == 'O'])
c_names = list(eye_final['file_name'][eye_final['labels'] == 'C'])
g_names = list(eye_final['file_name'][eye_final['labels'] == 'G'])
a_names = list(eye_final['file_name'][eye_final['labels'] == 'A'])
m_names = list(eye_final['file_name'][eye_final['labels'] == 'M'])
h_names = list(eye_final['file_name'][eye_final['labels'] == 'H'])


n_paths = list(eye_final['file_path'][eye_final['labels'] == 'N'])
d_paths = list(eye_final['file_path'][eye_final['labels'] == 'D'])
o_paths = list(eye_final['file_path'][eye_final['labels'] == 'O'])
c_paths = list(eye_final['file_path'][eye_final['labels'] == 'C'])
g_paths = list(eye_final['file_path'][eye_final['labels'] == 'G'])
a_paths = list(eye_final['file_path'][eye_final['labels'] == 'A'])
m_paths = list(eye_final['file_path'][eye_final['labels'] == 'M'])
```

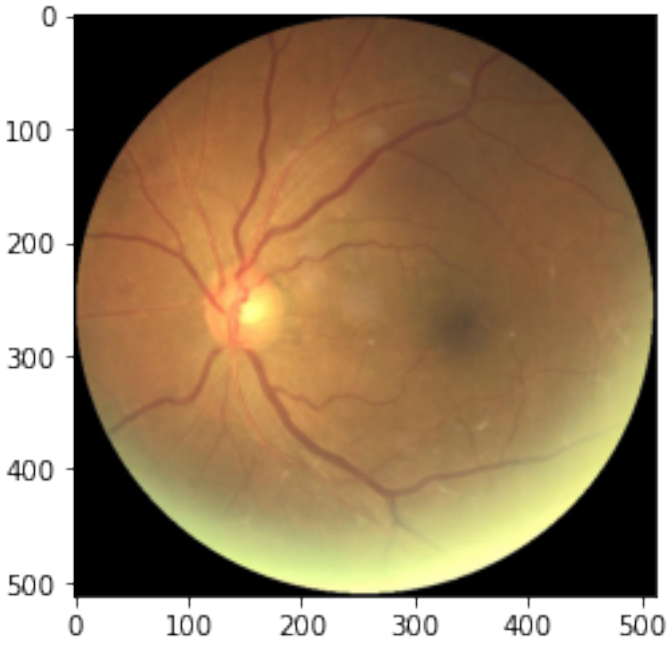
```
h_paths = list(eye_final['file_path'][eye_final['labels'] == 'H'])
```

[31]:
```
"""creating a function that copies images by category to new folders for data␣
 ↪augmentation
"""

def copy_to_folder(paths, category):
    for file in glob.glob('/Users/armaanvalvi/Documents/project-data/archive/
 ↪images/*.jpg'):
        if file in paths:
            shutil.copy(file, f'/Users/armaanvalvi/Documents/project-data/
 ↪archive/images_by_category/{category}')
```

[32]:
```
# copy_to_folder(a_paths, 'A')
# copy_to_folder(c_paths, 'C')
# copy_to_folder(d_paths, 'D')
# copy_to_folder(g_paths, 'G')
# copy_to_folder(h_paths, 'H')
# copy_to_folder(m_paths, 'M')
# copy_to_folder(n_paths, 'N')
# copy_to_folder(o_paths, 'O')
```

[33]:
```
import cv2
gf = cv2.imread('/Users/armaanvalvi/Documents/project-data/archive/images/
 ↪11_left.jpg')
dg = Image.open('/Users/armaanvalvi/Documents/project-data/archive/images/
 ↪11_left.jpg')
plt.imshow(dg)
```

[33]:
```
<matplotlib.image.AxesImage at 0x7faaade63450>
```



[34]:
```
""" taken from the keras blog site by founder Francois Chollet """

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

[35]:
```
""" creating a function using the datagen object for each of the image␣
 ↪categories using code from Keras"""

def get_more_images_2(category_input, save_input, prefix_input,␣
 ↪batch_size_input = 5, loop_input = 10):
    for file in glob.glob(f'/Users/armaanvalvi/Documents/project-data/archive/
 ↪images_by_category/{category_input}/*.jpg'):
        img = load_img(file)
        x = img_to_array(img)
        x = x.reshape((1,) + x.shape)
        i = 0
        for batch in datagen.flow(x, batch_size = batch_size_input,
```

```
                                save_to_dir = f'/Users/armaanvalvi/Documents/
↪project-data/archive/images_aug/{save_input}_aug',
                                save_prefix = f'{prefix_input}_aug',
                                save_format = 'jpg'):
            i += 1
            if i > loop_input:
                break
```

[36]:
```
# get_more_images_2('H', 'H_pilot', 'H', 1, 2)
# get_more_images_2('A', 'A', 'A')
# get_more_images_2('C', 'C', 'C')
# get_more_images_2('D', 'D', 'D', 1, 2)
# get_more_images_2('G', 'G', 'G')
# get_more_images_2('H', 'H', 'H', 5, 20)
# get_more_images_2('M', 'M', 'M')
# get_more_images_2('N', 'N', 'N', 1, 1)
# get_more_images_2('O', 'O', 'O', 2, 4)
```

[37]:
```
eye_final['labels'].value_counts()
```

[37]:
```
N    2873
D    1608
O     708
C     293
G     284
A     266
M     232
H     128
Name: labels, dtype: int64
```

[38]:
```
""" revisit for test_data """

# # os.makedirs('pilot_test')

# for file in glob.glob('/Users/armaanvalvi/Documents/project-data/archive/
↪images/*.jpg'):
#     if file in list(test_data['file_path']):
#         shutil.copy(file, '/Users/armaanvalvi/Documents/project-data/archive/
↪pilot_test')
```

[38]:
```
' revisit for test_data '
```

[39]:
```
if os.path.isdir('train_set/A') is False:
    os.makedirs('train_set/A')
    os.makedirs('train_set/C')
    os.makedirs('train_set/D')
    os.makedirs('train_set/G')
```

```
    os.makedirs('train_set/H')
    os.makedirs('train_set/M')
    os.makedirs('train_set/N')
    os.makedirs('train_set/O')

# randomly moving images to train_set
def move_to_train(category, category_input, number):
    for pic in random.sample(glob.glob(f'/Users/armaanvalvi/Documents/
 ↪project-data/archive/images_aug/{category}_aug/*.jpg'),
                             number):
        shutil.move(pic, f'/Users/armaanvalvi/Documents/project-data/archive/
 ↪train_set/{category_input}')
```

[40]:
```
# each folder had at least 2367 augmented images (H_aug had exactly 2367)

# move_to_train('A', 'A', 2367)
# move_to_train('C', 'C', 2367)
# move_to_train('D', 'D', 2367)
# move_to_train('G', 'G', 2367)
# move_to_train('H', 'H', 2367)
# move_to_train('M', 'M', 2367)
# move_to_train('N', 'N', 2367)
# move_to_train('O', 'O', 2367)
```

[41]:
```
train_batches = datagen.flow_from_directory(directory = '/Users/armaanvalvi/
 ↪Documents/project-data/archive/train_set/',
                                    target_size = (512,512),
                                    classes = ['A', 'C', 'D', 'G', 'H', 'M',␣
 ↪'N', 'O'],
                                    batch_size = 9)
```

```
Found 18936 images belonging to 8 classes.
```

[42]:
```
# path_of_input_image_classes = 'static/flower_photos'
# class_names = os.listdir(path_of_input_image_classes)
# num_of_output_classes = size(class_names)
# num_of_images_in_classes = []
# image_train = []

# for dir in class_names:
# filenames = os.listdir(os.path.join(path_of_input_image_classes,dir))
# num_of_images_in_classes.append(size(filenames))
#     for file in filenames:

#     #image = Image.open(os.path.join(path_of_input_image_classes, dir, file))
#     image = cv2.imread(os.path.join(path_of_input_image_classes, dir, file))
#     image = cv2.resize(image,(img_cols,img_rows))
```

```
#     image_train.append(image)
```

[43]:
```
train_imgs, train_labels = next(train_batches) # should be 9 images with the 9␣
↪corresponding labels
```

[44]:
```
# function to plot images in the form of a grid where images are placed
# from the tensorflow website

class_names = ['AMD', 'Cataract', 'Diabetes', 'Glaucoma', 'Hypertension',
               'Myopia', 'Normal', 'Other']

def plotImages(images_arr):
    fig, axes = plt.subplots(1,9, figsize=(20,20))
    axes = axes.flatten()
    for img, ax in zip(images_arr, axes):
        ax.imshow(img)
        ax.axis('off')
#         plt.xlabel(class_names[train_labels])
    plt.tight_layout()
    plt.show()
```

[45]:
```
plotImages(train_imgs)
print(train_labels)
```

```
[[1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1.]
 [0. 1. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 0. 0. 0. 1. 0. 0.]]
```

0.1.1 Build and Train a CNN

[46]:
```
# config = tf.compat.v1.ConfigProto()
# config.gpu_options.allow_growth = True
# # Now when creating your session pass this config to it.
# sess = tf.compat.v1.Session(config=config)
```

[47]:
```
model = Sequential([
    # the first hidden layer - Densely/fully connected layer
    Conv2D(filters = 24,
           kernel_size = (16,16),
           input_shape = (512,512,3), # creating an implicit input layer for␣
↪the model (height,width, no. of colors - RGB)
           activation = 'relu',
           kernel_initializer = 'glorot_uniform', # default value
           padding = 'same'), # zero padding to ensure the dimensionality of␣
↪the images isn't reduced after the convolution operations)
                              # otherwise, the default is padding = 'valid',␣
↪which does not pad the input for each indiv layer
    MaxPool2D(pool_size = (8,8), strides = 8, padding = 'valid'), # cut pooling␣
↪dimensions by half to shrink the number of trainable parameters
    Conv2D(filters = 48, # following common practice of increasing functions as␣
↪you have more hidden layers
           kernel_size = (16,16),
           activation = 'relu',
           padding = 'same',
           use_bias = True, # default = True
           bias_initializer = 'zeros'), # default = 'zeros'
    MaxPool2D(pool_size = (8,8), strides = 8),
    Flatten(), # before passing output from a convo layer to a dense layer, you␣
↪have to turn the output into a one-dim tensor by flattening the output by␣
↪multiplying the dimensions of the data from the conv layer by the filters in␣
↪that layer
    Dropout(0.3), # deactivate 30% of the neurons
    Dense(units = 8, activation = 'softmax')]) # the output layer
```

[48]:
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 512, 512, 24)      18456
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 64, 64, 24)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 64, 64, 48)        294960
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 8, 8, 48)          0
_________________________________________________________________
flatten (Flatten)            (None, 3072)              0
_________________________________________________________________
dropout (Dropout)            (None, 3072)              0
_________________________________________________________________
```

```
dense (Dense)                (None, 8)                 24584
=================================================================
Total params: 338,000
Trainable params: 338,000
Non-trainable params: 0
_________________________________________________________________
```

[49]:
```
# now that model is built, need to prepare it for training with model.compile

model.compile(optimizer=Adam(learning_rate = 0.01),
              loss = 'categorical_crossentropy',
              metrics = ['accuracy'
#                          , 'precision', 'recall', 'f1'
                         ])
```

[ ]:
```
# train the model with model.fit

model.fit(x = train_batches, validation_data = 0.15, epochs=1,
          verbose=0)
```

[ ]:
```

```