skin cancer project

January 24, 2020

Skin Cancer(HAM10000) Project Report Tahmeed Anjum Hossain 190520304

Table of Contents

1. Introduction
   Early detection of conditions of skin cancer is very crucial thus pigment skin diagnosis field has been growing rapidly. Recently various kinds of detection methods and the computerised algorithm is used to improve the accuracy of detection.

2. Resource and data
   The dataset is collected from a competition in Kaggle. It can be collected from here. Although the dataset was first made publicly available by the Harvard Database on June,2018. The main target was to provide training data so that the process of skin cancer lesion classification can be automated. If successful, the application can be helpful for medical professionals. It will also reduce cost and save time. Other than 10,015 images, a metadata file including demographic information of each lesion is provided as well. The 7 classes of skin cancer lesions included in this dataset are:

Melanocytic nevi

`<li>Melanoma </li>`

Benign keratosis-like lesions

Basal cell carcinoma

Actinic keratoses

Vascular lesions

Dermatofibroma

3. Project objective & planning

For this specific dataset, computation time is more important than getting a higher accuracy. As for cancer, it is crucial to detect the disease at early stage, which means dataset of large population need to be processed as quickly as possible. In this primary level, accuracy is calculated between lower resolution model and higher resolution by keeping the computation time factor in mind.

4. Exploratory data analysis
   At first necessary library is installed for the project.

[320]:
```
import numpy as np
import pandas as pd
import imageio
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, precision_recall_fscore_support
import scipy.ndimage
from scipy import misc
import skimage
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
plt.style.use('fivethirtyeight')

from tqdm import tqdm
from glob import glob
from scipy import stats
from sklearn.preprocessing import LabelEncoder, StandardScaler

!pip install keras
import keras

from PIL import Image
```

```
Requirement already satisfied: keras in /opt/anaconda3/lib/python3.7/site-
packages (2.3.1)
Requirement already satisfied: keras-applications>=1.0.6 in
/opt/anaconda3/lib/python3.7/site-packages (from keras) (1.0.8)
Requirement already satisfied: numpy>=1.9.1 in
/opt/anaconda3/lib/python3.7/site-packages (from keras) (1.17.2)
Requirement already satisfied: keras-preprocessing>=1.0.5 in
/opt/anaconda3/lib/python3.7/site-packages (from keras) (1.1.0)
Requirement already satisfied: scipy>=0.14 in /opt/anaconda3/lib/python3.7/site-
packages (from keras) (1.4.1)
Requirement already satisfied: h5py in /opt/anaconda3/lib/python3.7/site-
packages (from keras) (2.9.0)
Requirement already satisfied: six>=1.9.0 in /opt/anaconda3/lib/python3.7/site-
packages (from keras) (1.12.0)
Requirement already satisfied: pyyaml in /opt/anaconda3/lib/python3.7/site-
packages (from keras) (5.1.2)
```

[321]:
```python
!pip install tensorflow
import tensorflow as tf
from keras.utils import to_categorical
from keras.layers import Dense, Input, Flatten, Reshape, Conv2D, MaxPool2D,␣
 ↪concatenate, Activation, Dropout
from keras.optimizers import Adam, RMSprop
from keras.models import Model, Sequential, load_model
from keras.losses import binary_crossentropy
from keras.metrics import binary_accuracy
from keras.callbacks import ModelCheckpoint
from keras.preprocessing.image import ImageDataGenerator
```

```
Requirement already satisfied: tensorflow in /opt/anaconda3/lib/python3.7/site-
packages (2.1.0)
Requirement already satisfied: wrapt>=1.11.1 in
/opt/anaconda3/lib/python3.7/site-packages (from tensorflow) (1.11.2)
Requirement already satisfied: wheel>=0.26; python_version >= "3" in
/opt/anaconda3/lib/python3.7/site-packages (from tensorflow) (0.33.6)
Requirement already satisfied: termcolor>=1.1.0 in
/opt/anaconda3/lib/python3.7/site-packages (from tensorflow) (1.1.0)
Requirement already satisfied: keras-applications>=1.0.8 in
/opt/anaconda3/lib/python3.7/site-packages (from tensorflow) (1.0.8)
Requirement already satisfied: absl-py>=0.7.0 in
/opt/anaconda3/lib/python3.7/site-packages (from tensorflow) (0.9.0)
Requirement already satisfied: tensorboard<2.2.0,>=2.1.0 in
/opt/anaconda3/lib/python3.7/site-packages (from tensorflow) (2.1.0)
Requirement already satisfied: astor>=0.6.0 in
/opt/anaconda3/lib/python3.7/site-packages (from tensorflow) (0.8.1)
Requirement already satisfied: gast==0.2.2 in /opt/anaconda3/lib/python3.7/site-
packages (from tensorflow) (0.2.2)
Requirement already satisfied: numpy<2.0,>=1.16.0 in
/opt/anaconda3/lib/python3.7/site-packages (from tensorflow) (1.17.2)
Requirement already satisfied: six>=1.12.0 in /opt/anaconda3/lib/python3.7/site-
packages (from tensorflow) (1.12.0)
Requirement already satisfied: google-pasta>=0.1.6 in
/opt/anaconda3/lib/python3.7/site-packages (from tensorflow) (0.1.8)
Requirement already satisfied: grpcio>=1.8.6 in
/opt/anaconda3/lib/python3.7/site-packages (from tensorflow) (1.26.0)
Requirement already satisfied: scipy==1.4.1; python_version >= "3" in
/opt/anaconda3/lib/python3.7/site-packages (from tensorflow) (1.4.1)
Requirement already satisfied: keras-preprocessing>=1.1.0 in
/opt/anaconda3/lib/python3.7/site-packages (from tensorflow) (1.1.0)
Requirement already satisfied: protobuf>=3.8.0 in
/opt/anaconda3/lib/python3.7/site-packages (from tensorflow) (3.11.2)
Requirement already satisfied: opt-einsum>=2.3.2 in
/opt/anaconda3/lib/python3.7/site-packages (from tensorflow) (3.1.0)
Requirement already satisfied: tensorflow-estimator<2.2.0,>=2.1.0rc0 in
```

```
/opt/anaconda3/lib/python3.7/site-packages (from tensorflow) (2.1.0)
Requirement already satisfied: h5py in /opt/anaconda3/lib/python3.7/site-
packages (from keras-applications>=1.0.8->tensorflow) (2.9.0)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in
/opt/anaconda3/lib/python3.7/site-packages (from
tensorboard<2.2.0,>=2.1.0->tensorflow) (0.4.1)
Requirement already satisfied: markdown>=2.6.8 in
/opt/anaconda3/lib/python3.7/site-packages (from
tensorboard<2.2.0,>=2.1.0->tensorflow) (3.1.1)
Requirement already satisfied: werkzeug>=0.11.15 in
/opt/anaconda3/lib/python3.7/site-packages (from
tensorboard<2.2.0,>=2.1.0->tensorflow) (0.16.0)
Requirement already satisfied: requests<3,>=2.21.0 in
/opt/anaconda3/lib/python3.7/site-packages (from
tensorboard<2.2.0,>=2.1.0->tensorflow) (2.22.0)
Requirement already satisfied: setuptools>=41.0.0 in
/opt/anaconda3/lib/python3.7/site-packages (from
tensorboard<2.2.0,>=2.1.0->tensorflow) (41.4.0)
Requirement already satisfied: google-auth<2,>=1.6.3 in
/opt/anaconda3/lib/python3.7/site-packages (from
tensorboard<2.2.0,>=2.1.0->tensorflow) (1.10.1)
Requirement already satisfied: requests-oauthlib>=0.7.0 in
/opt/anaconda3/lib/python3.7/site-packages (from google-auth-
oauthlib<0.5,>=0.4.1->tensorboard<2.2.0,>=2.1.0->tensorflow) (1.3.0)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in
/opt/anaconda3/lib/python3.7/site-packages (from
requests<3,>=2.21.0->tensorboard<2.2.0,>=2.1.0->tensorflow) (3.0.4)
Requirement already satisfied: certifi>=2017.4.17 in
/opt/anaconda3/lib/python3.7/site-packages (from
requests<3,>=2.21.0->tensorboard<2.2.0,>=2.1.0->tensorflow) (2019.9.11)
Requirement already satisfied: idna<2.9,>=2.5 in
/opt/anaconda3/lib/python3.7/site-packages (from
requests<3,>=2.21.0->tensorboard<2.2.0,>=2.1.0->tensorflow) (2.8)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
/opt/anaconda3/lib/python3.7/site-packages (from
requests<3,>=2.21.0->tensorboard<2.2.0,>=2.1.0->tensorflow) (1.24.2)
Requirement already satisfied: cachetools<5.0,>=2.0.0 in
/opt/anaconda3/lib/python3.7/site-packages (from google-
auth<2,>=1.6.3->tensorboard<2.2.0,>=2.1.0->tensorflow) (4.0.0)
Requirement already satisfied: rsa<4.1,>=3.1.4 in
/opt/anaconda3/lib/python3.7/site-packages (from google-
auth<2,>=1.6.3->tensorboard<2.2.0,>=2.1.0->tensorflow) (4.0)
Requirement already satisfied: pyasn1-modules>=0.2.1 in
/opt/anaconda3/lib/python3.7/site-packages (from google-
auth<2,>=1.6.3->tensorboard<2.2.0,>=2.1.0->tensorflow) (0.2.8)
Requirement already satisfied: oauthlib>=3.0.0 in
/opt/anaconda3/lib/python3.7/site-packages (from requests-
oauthlib>=0.7.0->google-auth-
oauthlib<0.5,>=0.4.1->tensorboard<2.2.0,>=2.1.0->tensorflow) (3.1.0)
Requirement already satisfied: pyasn1>=0.1.3 in
/opt/anaconda3/lib/python3.7/site-packages (from rsa<4.1,>=3.1.4->google-
auth<2,>=1.6.3->tensorboard<2.2.0,>=2.1.0->tensorflow) (0.4.8)
```

[322]:
```
!pip install imblearn
```

```
Requirement already satisfied: imblearn in /opt/anaconda3/lib/python3.7/site-
packages (0.0)
Requirement already satisfied: imbalanced-learn in
/opt/anaconda3/lib/python3.7/site-packages (from imblearn) (0.6.1)
Requirement already satisfied: scipy>=0.17 in /opt/anaconda3/lib/python3.7/site-
packages (from imbalanced-learn->imblearn) (1.4.1)
Requirement already satisfied: joblib>=0.11 in
/opt/anaconda3/lib/python3.7/site-packages (from imbalanced-learn->imblearn)
(0.13.2)
Requirement already satisfied: scikit-learn>=0.22 in
/opt/anaconda3/lib/python3.7/site-packages (from imbalanced-learn->imblearn)
(0.22.1)
Requirement already satisfied: numpy>=1.11 in /opt/anaconda3/lib/python3.7/site-
packages (from imbalanced-learn->imblearn) (1.17.2)
```

[323]:
```
!pip install scipy
```

```
Requirement already satisfied: scipy in /opt/anaconda3/lib/python3.7/site-
packages (1.4.1)
Requirement already satisfied: numpy>=1.13.3 in
/opt/anaconda3/lib/python3.7/site-packages (from scipy) (1.17.2)
```

[324]:
```
from imblearn.under_sampling import RandomUnderSampler
```

EDA process starts from here by loading the data in the jupyter notebook

[325]:
```
# importing data
metadata = pd.read_csv('/Users/ragnar/Downloads/skin-cancer-mnist-ham10000/
 ↪HAM10000_metadata.csv')
```

[326]:
```
print(metadata.shape)
```

```
(10015, 7)
```

[327]:
```
# encoded label for all the seven class

le = LabelEncoder()
le.fit(metadata['dx'])
LabelEncoder()
print(list(le.classes_))
```

```
metadata['label'] = le.transform(metadata["dx"])
metadata.sample(10)
```

```
['akiec', 'bcc', 'bkl', 'df', 'mel', 'nv', 'vasc']
```

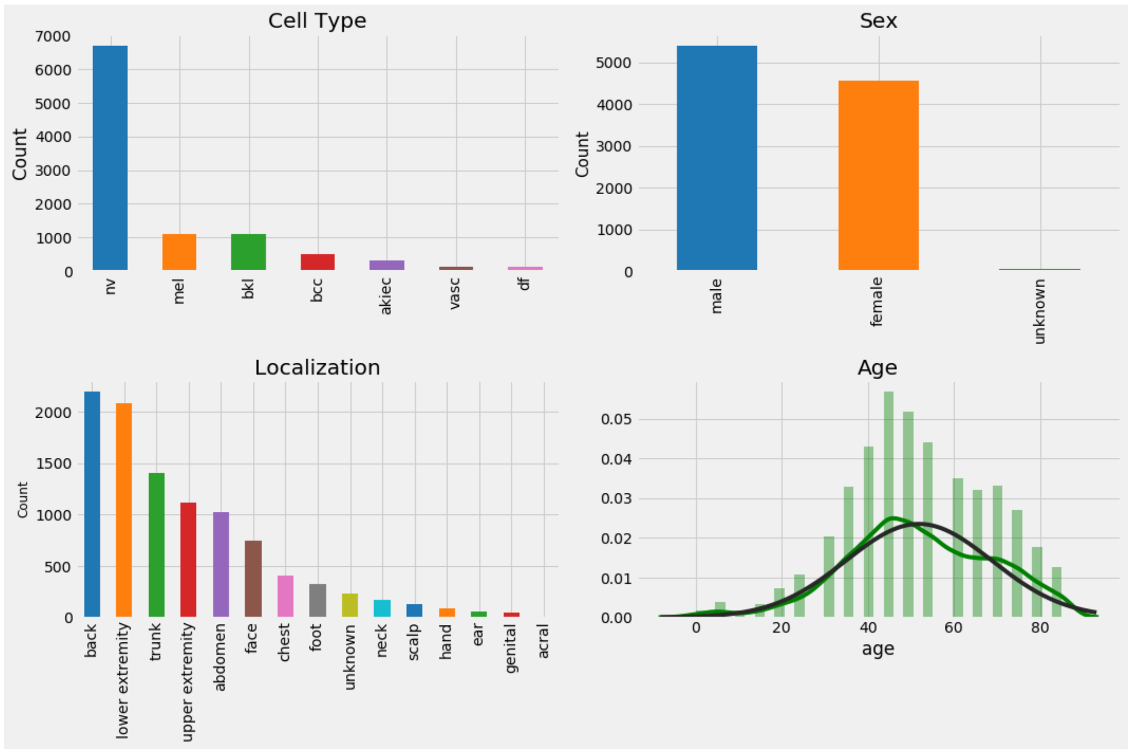
[327]:

```
        lesion_id      image_id     dx    dx_type   age     sex  \
5844  HAM_0000400  ISIC_0031414     nv  follow_up  45.0    male
9853  HAM_0005650  ISIC_0030387  akiec      histo  65.0    male
2261  HAM_0002131  ISIC_0030901    mel      histo  55.0    male
7428  HAM_0006857  ISIC_0033475     nv      histo  40.0  female
3302  HAM_0006845  ISIC_0030170     nv  follow_up  45.0  female
2855  HAM_0006588  ISIC_0031552    bcc      histo  70.0  female
9687  HAM_0002644  ISIC_0029417  akiec      histo  80.0  female
1001  HAM_0003668  ISIC_0024383    bkl  consensus  70.0  female
8578  HAM_0003562  ISIC_0030544     nv      histo  60.0  female
3103  HAM_0000408  ISIC_0027000     nv  follow_up  45.0  female

         localization  label
5844  upper extremity      5
9853  lower extremity      0
2261  upper extremity      4
7428             back      5
3302  upper extremity      5
2855             face      1
9687             neck      0
1001            trunk      2
8578             face      5
3103  lower extremity      5
```

[328]:

```
# glimpse of distribution from each column

fig = plt.figure(figsize=(15,10))

colors = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd',
          '#8c564b', '#e377c2', '#7f7f7f', '#bcbd22', '#17becf']

g1 = fig.add_subplot(221)
metadata['dx'].value_counts().plot(kind='bar', ax=g1, color=colors)
g1.set_ylabel('Count')
g1.set_title('Cell Type');

g2 = fig.add_subplot(222)
metadata['sex'].value_counts().plot(kind='bar', ax=g2, color=colors)
g2.set_ylabel('Count', size=15)
g2.set_title('Sex');

g3 = fig.add_subplot(223)
```

```
metadata['localization'].value_counts().plot(kind='bar', color=colors)
g3.set_ylabel('Count',size=12)
g3.set_title('Localization')


g4 = fig.add_subplot(224)
sample_age = metadata[pd.notnull(metadata['age'])]
sns.distplot(sample_age['age'], fit=stats.norm, color='green');
g4.set_title('Age')

plt.tight_layout()
plt.show()
```



5. Image processing
   At first, 28x28x3 images are analysed.

```
[329]: image = pd.read_csv('/Users/ragnar/Downloads/skin-cancer-mnist-ham10000/
        ↪hmnist_28_28_RGB.csv', header=0)
       featured_columns = image.columns[:-1]
       featured_columns
       image = image.drop(image.index[0])
```

```
[330]: display(image.shape, image.head())
```

```
(10014, 2353)
```

```
   pixel0000  pixel0001  pixel0002  pixel0003  pixel0004  pixel0005  \
1         25         14         30         68         48         75
2        192        138        153        200        145        163
3         38         19         30         95         59         72
4        158        113        139        194        144        174
5          8          1          3         19          5         10

   pixel0006  pixel0007  pixel0008  pixel0009  ...  pixel2343  pixel2344  \
1        123         93        126        158  ...         60         39
2        201        142        160        206  ...        167        129
3        143        103        119        171  ...         44         26
4        215        162        191        225  ...        209        166
5         26          8         13         34  ...         12          3

   pixel2345  pixel2346  pixel2347  pixel2348  pixel2349  pixel2350  \
1         55         25         14         28         25         14
2        143        159        124        142        136        104
3         36         25         12         17         25         12
4        185        172        135        149        109         78
5          7          5          0          1          4          0

   pixel2351  label
1         27      2
2        117      2
3         15      2
4         92      2
5          0      2

[5 rows x 2353 columns]
```

[331]:
```
# variable x & y are assigned for modelling
x = image[featured_columns]
y = image.label
```

[332]:
```
x = np.asarray(x)/225
x = x.reshape(-1,28,28,3)
y = np.asarray(y)
print(x.shape, y.shape)
```

```
(10014, 28, 28, 3) (10014,)
```

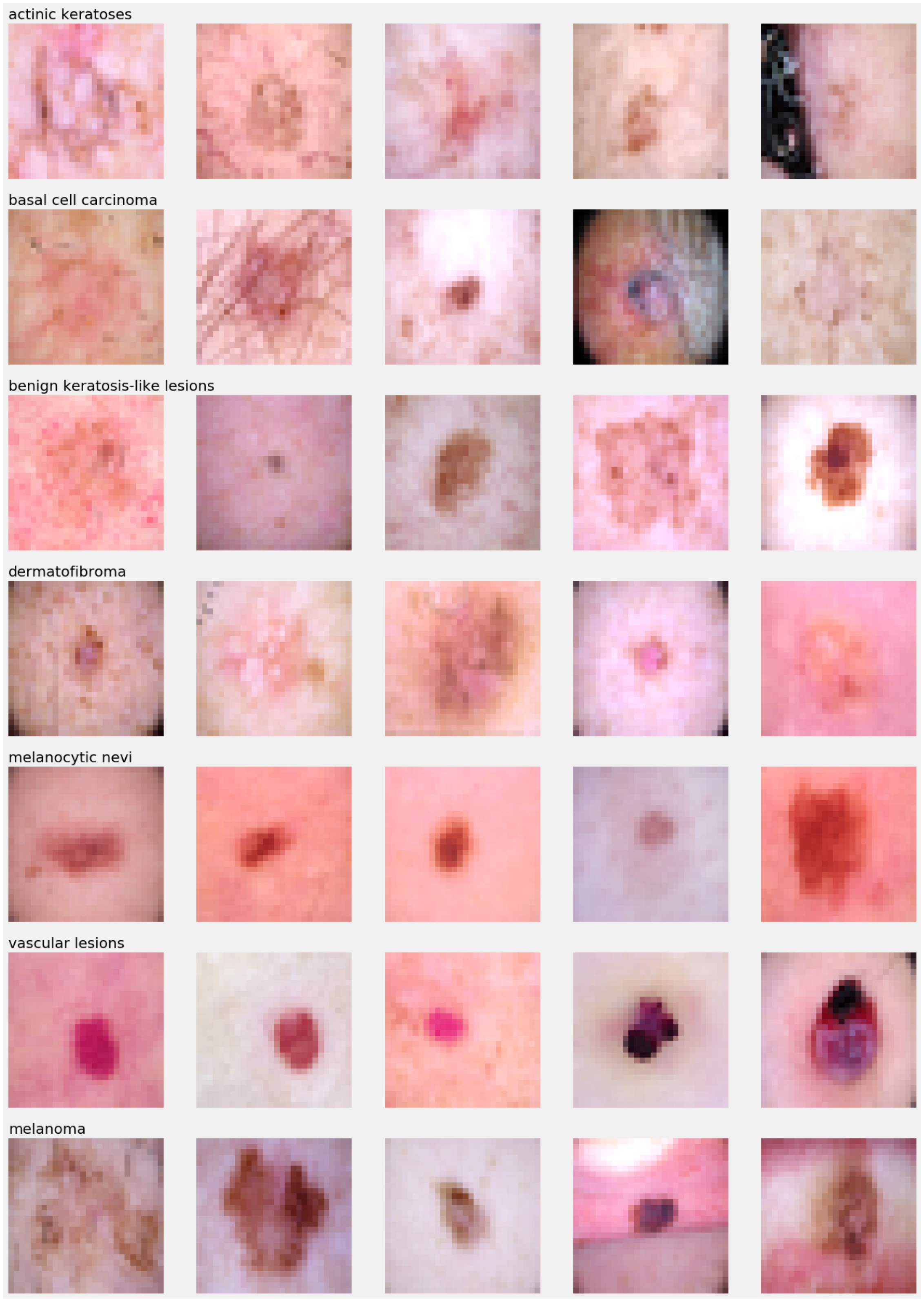
[333]:
```
# split into test dataset and train dataset
```

```python
x_train_s, x_test_s, y_train_s, y_test_s = train_test_split(x, y, test_size=0.
 ↪4, stratify=y)
print(x_train_s.shape)
print(y_train_s.shape)
print(x_test_s.shape)
print(y_test_s.shape)
```

```
(6008, 28, 28, 3)
(6008,)
(4006, 28, 28, 3)
(4006,)
```

[334]:
```python
fig = plt.figure(figsize=(20, 30))
j=0
names = ['actinic keratoses', 'basal cell carcinoma', 'benign keratosis-like␣
 ↪lesions','dermatofibroma',
         'melanocytic nevi','vascular lesions','melanoma']

for class_number in range(7):

    example = x_train_s[np.where(y_train_s == class_number)][:5]

    for i, picture in enumerate(example):
        n=i+j
        picture.reshape(28,28,3)

        plt.subplot(7,5,n+1)
        plt.imshow(picture)
        if i ==0:
            plt.title(f'{names[class_number]}', size=20, loc='left')

        plt.axis('off')
    j+=5
plt.show()
fig.savefig('small_pix_sample.jpg')
```

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
```


actinic keratoses
basal cell carcinoma
benign keratosis-like lesions
dermatofibroma
melanocytic nevi
vascular lesions
melanoma

Running a simple architecture to see the baseline accuracy with smaller picture

[335]:
```
model = Sequential()

model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(28,28,3)))
model.add(MaxPool2D((2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPool2D((2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPool2D((2, 2)))

model.add(Flatten())

model.add(Dense(512, activation='relu'))
model.add(Dense(7))
model.add(Activation(tf.nn.softmax))


print(model.summary())
```

```
Model: "sequential_15"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_50 (Conv2D)           (None, 26, 26, 32)        896
_________________________________________________________________
max_pooling2d_40 (MaxPooling (None, 13, 13, 32)        0
_________________________________________________________________
conv2d_51 (Conv2D)           (None, 11, 11, 64)        18496
_________________________________________________________________
max_pooling2d_41 (MaxPooling (None, 5, 5, 64)          0
_________________________________________________________________
conv2d_52 (Conv2D)           (None, 3, 3, 128)         73856
_________________________________________________________________
max_pooling2d_42 (MaxPooling (None, 1, 1, 128)         0
_________________________________________________________________
flatten_15 (Flatten)         (None, 128)               0
_________________________________________________________________
dense_32 (Dense)             (None, 512)               66048
_________________________________________________________________
dense_33 (Dense)             (None, 7)                 3591
_________________________________________________________________
activation_8 (Activation)    (None, 7)                 0
=================================================================
Total params: 162,887
Trainable params: 162,887
Non-trainable params: 0
_________________________________________________________________
```

```
None
```

[336]:
```
#training model size: 28x28x3

y_test_s = to_categorical(y_test_s, num_classes=7)
y_train_s = to_categorical(y_train_s, num_classes=7)
```

[337]:
```
model.compile(loss='categorical_crossentropy',
optimizer='rmsprop',
metrics=['acc'])

history = model.fit(
    x_train_s, y_train_s,
    epochs=150,
    batch_size = 128,
    validation_data=(x_test_s, y_test_s),
    verbose=2)
```

```
Train on 6008 samples, validate on 4006 samples
Epoch 1/150
 - 2s - loss: 1.1881 - acc: 0.6566 - val_loss: 1.0159 - val_acc: 0.6695
Epoch 2/150
 - 2s - loss: 1.0100 - acc: 0.6696 - val_loss: 0.9906 - val_acc: 0.6695
Epoch 3/150
 - 2s - loss: 0.9519 - acc: 0.6714 - val_loss: 0.9178 - val_acc: 0.6697
Epoch 4/150
 - 2s - loss: 0.9243 - acc: 0.6681 - val_loss: 0.8798 - val_acc: 0.6792
Epoch 5/150
 - 2s - loss: 0.8983 - acc: 0.6841 - val_loss: 0.8536 - val_acc: 0.7007
Epoch 6/150
 - 2s - loss: 0.8963 - acc: 0.6791 - val_loss: 0.8585 - val_acc: 0.7047
Epoch 7/150
 - 2s - loss: 0.8704 - acc: 0.6904 - val_loss: 0.8243 - val_acc: 0.6997
Epoch 8/150
 - 2s - loss: 0.8527 - acc: 0.6921 - val_loss: 0.8226 - val_acc: 0.7127
Epoch 9/150
 - 2s - loss: 0.8338 - acc: 0.6939 - val_loss: 0.8536 - val_acc: 0.6875
Epoch 10/150
 - 2s - loss: 0.8236 - acc: 0.6972 - val_loss: 0.8009 - val_acc: 0.7254
Epoch 11/150
 - 2s - loss: 0.8045 - acc: 0.7044 - val_loss: 0.7900 - val_acc: 0.7079
Epoch 12/150
 - 2s - loss: 0.7804 - acc: 0.7164 - val_loss: 0.7588 - val_acc: 0.7309
Epoch 13/150
 - 2s - loss: 0.7602 - acc: 0.7209 - val_loss: 0.7665 - val_acc: 0.7177
Epoch 14/150
 - 2s - loss: 0.7501 - acc: 0.7209 - val_loss: 0.7158 - val_acc: 0.7339
Epoch 15/150
```

```
 - 2s - loss: 0.7413 - acc: 0.7289 - val_loss: 0.7457 - val_acc: 0.7142
Epoch 16/150
 - 2s - loss: 0.7228 - acc: 0.7349 - val_loss: 0.7768 - val_acc: 0.7069
Epoch 17/150
 - 2s - loss: 0.7232 - acc: 0.7354 - val_loss: 0.7193 - val_acc: 0.7302
Epoch 18/150
 - 2s - loss: 0.7118 - acc: 0.7347 - val_loss: 0.7274 - val_acc: 0.7289
Epoch 19/150
 - 2s - loss: 0.6951 - acc: 0.7422 - val_loss: 0.7190 - val_acc: 0.7411
Epoch 20/150
 - 2s - loss: 0.6802 - acc: 0.7455 - val_loss: 0.7966 - val_acc: 0.7054
Epoch 21/150
 - 2s - loss: 0.6724 - acc: 0.7518 - val_loss: 0.6956 - val_acc: 0.7479
Epoch 22/150
 - 2s - loss: 0.6605 - acc: 0.7565 - val_loss: 0.6934 - val_acc: 0.7504
Epoch 23/150
 - 2s - loss: 0.6481 - acc: 0.7595 - val_loss: 0.8682 - val_acc: 0.6930
Epoch 24/150
 - 2s - loss: 0.6448 - acc: 0.7572 - val_loss: 0.7227 - val_acc: 0.7207
Epoch 25/150
 - 2s - loss: 0.6233 - acc: 0.7656 - val_loss: 0.6990 - val_acc: 0.7421
Epoch 26/150
 - 2s - loss: 0.6083 - acc: 0.7695 - val_loss: 0.8100 - val_acc: 0.7284
Epoch 27/150
 - 2s - loss: 0.6225 - acc: 0.7706 - val_loss: 0.7445 - val_acc: 0.7371
Epoch 28/150
 - 2s - loss: 0.5868 - acc: 0.7853 - val_loss: 0.7070 - val_acc: 0.7341
Epoch 29/150
 - 2s - loss: 0.5976 - acc: 0.7761 - val_loss: 0.6794 - val_acc: 0.7491
Epoch 30/150
 - 2s - loss: 0.5817 - acc: 0.7788 - val_loss: 0.7938 - val_acc: 0.7257
Epoch 31/150
 - 2s - loss: 0.5508 - acc: 0.7958 - val_loss: 0.7073 - val_acc: 0.7406
Epoch 32/150
 - 2s - loss: 0.5611 - acc: 0.7879 - val_loss: 0.6580 - val_acc: 0.7649
Epoch 33/150
 - 2s - loss: 0.5516 - acc: 0.7941 - val_loss: 0.6787 - val_acc: 0.7551
Epoch 34/150
 - 2s - loss: 0.5270 - acc: 0.8023 - val_loss: 0.8708 - val_acc: 0.7192
Epoch 35/150
 - 2s - loss: 0.5220 - acc: 0.8116 - val_loss: 0.7031 - val_acc: 0.7464
Epoch 36/150
 - 2s - loss: 0.4969 - acc: 0.8154 - val_loss: 0.8012 - val_acc: 0.7099
Epoch 37/150
 - 2s - loss: 0.5075 - acc: 0.8132 - val_loss: 0.7872 - val_acc: 0.7304
Epoch 38/150
 - 2s - loss: 0.4922 - acc: 0.8164 - val_loss: 0.7166 - val_acc: 0.7441
Epoch 39/150
```

```
 - 2s - loss: 0.4652 - acc: 0.8261 - val_loss: 0.9061 - val_acc: 0.7164
Epoch 40/150
 - 2s - loss: 0.4566 - acc: 0.8314 - val_loss: 0.7253 - val_acc: 0.7421
Epoch 41/150
 - 2s - loss: 0.4528 - acc: 0.8357 - val_loss: 0.7201 - val_acc: 0.7524
Epoch 42/150
 - 2s - loss: 0.4324 - acc: 0.8387 - val_loss: 0.8231 - val_acc: 0.6910
Epoch 43/150
 - 2s - loss: 0.4284 - acc: 0.8399 - val_loss: 0.7223 - val_acc: 0.7444
Epoch 44/150
 - 2s - loss: 0.4021 - acc: 0.8525 - val_loss: 0.8228 - val_acc: 0.7114
Epoch 45/150
 - 2s - loss: 0.3860 - acc: 0.8640 - val_loss: 0.7992 - val_acc: 0.7409
Epoch 46/150
 - 2s - loss: 0.3800 - acc: 0.8648 - val_loss: 0.8367 - val_acc: 0.7289
Epoch 47/150
 - 2s - loss: 0.3657 - acc: 0.8678 - val_loss: 0.8184 - val_acc: 0.7454
Epoch 48/150
 - 2s - loss: 0.3501 - acc: 0.8702 - val_loss: 1.0506 - val_acc: 0.6638
Epoch 49/150
 - 2s - loss: 0.3525 - acc: 0.8685 - val_loss: 0.8376 - val_acc: 0.7209
Epoch 50/150
 - 2s - loss: 0.3269 - acc: 0.8742 - val_loss: 0.9414 - val_acc: 0.7364
Epoch 51/150
 - 2s - loss: 0.3292 - acc: 0.8813 - val_loss: 0.8066 - val_acc: 0.7541
Epoch 52/150
 - 2s - loss: 0.3102 - acc: 0.8847 - val_loss: 0.8129 - val_acc: 0.7481
Epoch 53/150
 - 2s - loss: 0.2872 - acc: 0.8955 - val_loss: 0.8473 - val_acc: 0.7501
Epoch 54/150
 - 2s - loss: 0.2736 - acc: 0.8980 - val_loss: 1.0994 - val_acc: 0.7194
Epoch 55/150
 - 2s - loss: 0.2867 - acc: 0.8978 - val_loss: 0.9106 - val_acc: 0.7327
Epoch 56/150
 - 2s - loss: 0.2527 - acc: 0.9111 - val_loss: 0.8894 - val_acc: 0.7489
Epoch 57/150
 - 2s - loss: 0.2605 - acc: 0.9041 - val_loss: 0.8629 - val_acc: 0.7411
Epoch 58/150
 - 2s - loss: 0.2463 - acc: 0.9100 - val_loss: 0.9574 - val_acc: 0.7576
Epoch 59/150
 - 2s - loss: 0.2436 - acc: 0.9131 - val_loss: 0.8792 - val_acc: 0.7399
Epoch 60/150
 - 2s - loss: 0.2043 - acc: 0.9256 - val_loss: 0.9643 - val_acc: 0.7389
Epoch 61/150
 - 2s - loss: 0.2218 - acc: 0.9216 - val_loss: 0.9316 - val_acc: 0.7374
Epoch 62/150
 - 2s - loss: 0.2042 - acc: 0.9261 - val_loss: 0.9427 - val_acc: 0.7424
Epoch 63/150
```

```
 - 2s - loss: 0.2027 - acc: 0.9288 - val_loss: 1.0355 - val_acc: 0.7356
Epoch 64/150
 - 2s - loss: 0.1798 - acc: 0.9334 - val_loss: 1.2126 - val_acc: 0.6692
Epoch 65/150
 - 2s - loss: 0.1945 - acc: 0.9281 - val_loss: 0.9784 - val_acc: 0.7496
Epoch 66/150
 - 2s - loss: 0.1647 - acc: 0.9466 - val_loss: 1.1463 - val_acc: 0.7312
Epoch 67/150
 - 2s - loss: 0.1606 - acc: 0.9412 - val_loss: 1.1938 - val_acc: 0.7309
Epoch 68/150
 - 2s - loss: 0.1632 - acc: 0.9522 - val_loss: 1.1328 - val_acc: 0.7506
Epoch 69/150
 - 2s - loss: 0.1443 - acc: 0.9529 - val_loss: 1.1955 - val_acc: 0.7469
Epoch 70/150
 - 2s - loss: 0.1838 - acc: 0.9486 - val_loss: 1.0957 - val_acc: 0.7391
Epoch 71/150
 - 2s - loss: 0.1332 - acc: 0.9532 - val_loss: 1.4575 - val_acc: 0.7334
Epoch 72/150
 - 2s - loss: 0.1300 - acc: 0.9557 - val_loss: 1.2543 - val_acc: 0.7466
Epoch 73/150
 - 2s - loss: 0.1306 - acc: 0.9572 - val_loss: 1.6562 - val_acc: 0.7204
Epoch 74/150
 - 2s - loss: 0.1235 - acc: 0.9645 - val_loss: 1.4651 - val_acc: 0.6855
Epoch 75/150
 - 2s - loss: 0.1341 - acc: 0.9571 - val_loss: 1.2945 - val_acc: 0.7456
Epoch 76/150
 - 2s - loss: 0.1446 - acc: 0.9664 - val_loss: 1.8332 - val_acc: 0.6303
Epoch 77/150
 - 2s - loss: 0.1386 - acc: 0.9574 - val_loss: 1.2490 - val_acc: 0.7379
Epoch 78/150
 - 2s - loss: 0.0914 - acc: 0.9737 - val_loss: 1.4847 - val_acc: 0.7272
Epoch 79/150
 - 2s - loss: 0.1582 - acc: 0.9566 - val_loss: 1.2257 - val_acc: 0.7451
Epoch 80/150
 - 2s - loss: 0.1202 - acc: 0.9632 - val_loss: 1.2656 - val_acc: 0.7416
Epoch 81/150
 - 2s - loss: 0.1062 - acc: 0.9725 - val_loss: 1.2543 - val_acc: 0.7476
Epoch 82/150
 - 2s - loss: 0.1086 - acc: 0.9665 - val_loss: 1.2573 - val_acc: 0.7441
Epoch 83/150
 - 2s - loss: 0.1117 - acc: 0.9747 - val_loss: 1.3322 - val_acc: 0.7499
Epoch 84/150
 - 2s - loss: 0.1749 - acc: 0.9587 - val_loss: 1.2547 - val_acc: 0.7292
Epoch 85/150
 - 2s - loss: 0.0817 - acc: 0.9820 - val_loss: 1.3730 - val_acc: 0.7491
Epoch 86/150
 - 2s - loss: 0.0904 - acc: 0.9725 - val_loss: 2.4167 - val_acc: 0.6950
Epoch 87/150
```

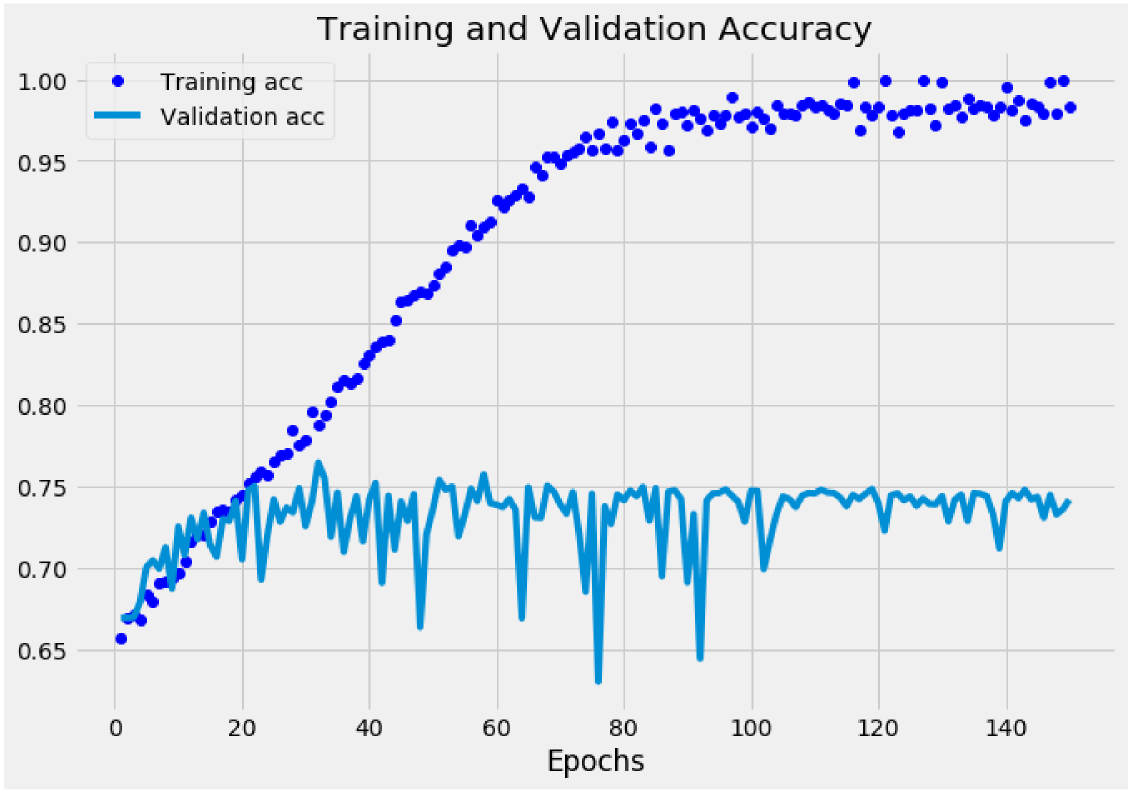
```
 - 2s - loss: 0.1841 - acc: 0.9569 - val_loss: 1.3026 - val_acc: 0.7469
Epoch 88/150
 - 2s - loss: 0.0875 - acc: 0.9792 - val_loss: 1.3555 - val_acc: 0.7479
Epoch 89/150
 - 2s - loss: 0.1015 - acc: 0.9797 - val_loss: 1.4026 - val_acc: 0.7424
Epoch 90/150
 - 2s - loss: 0.1174 - acc: 0.9717 - val_loss: 1.5374 - val_acc: 0.6912
Epoch 91/150
 - 2s - loss: 0.0632 - acc: 0.9812 - val_loss: 1.5049 - val_acc: 0.7332
Epoch 92/150
 - 2s - loss: 0.1045 - acc: 0.9757 - val_loss: 2.1146 - val_acc: 0.6445
Epoch 93/150
 - 2s - loss: 0.1338 - acc: 0.9689 - val_loss: 1.4093 - val_acc: 0.7414
Epoch 94/150
 - 2s - loss: 0.0855 - acc: 0.9785 - val_loss: 1.4576 - val_acc: 0.7459
Epoch 95/150
 - 2s - loss: 0.1005 - acc: 0.9725 - val_loss: 1.4012 - val_acc: 0.7461
Epoch 96/150
 - 2s - loss: 0.0937 - acc: 0.9779 - val_loss: 1.4996 - val_acc: 0.7484
Epoch 97/150
 - 2s - loss: 0.0422 - acc: 0.9897 - val_loss: 1.6314 - val_acc: 0.7444
Epoch 98/150
 - 2s - loss: 0.0923 - acc: 0.9770 - val_loss: 1.6147 - val_acc: 0.7406
Epoch 99/150
 - 2s - loss: 0.0902 - acc: 0.9792 - val_loss: 1.6162 - val_acc: 0.7284
Epoch 100/150
 - 2s - loss: 0.1301 - acc: 0.9710 - val_loss: 1.5783 - val_acc: 0.7476
Epoch 101/150
 - 2s - loss: 0.0960 - acc: 0.9799 - val_loss: 1.6324 - val_acc: 0.7474
Epoch 102/150
 - 2s - loss: 0.0935 - acc: 0.9760 - val_loss: 3.1619 - val_acc: 0.6995
Epoch 103/150
 - 2s - loss: 0.1970 - acc: 0.9697 - val_loss: 1.6578 - val_acc: 0.7177
Epoch 104/150
 - 2s - loss: 0.0640 - acc: 0.9847 - val_loss: 1.6303 - val_acc: 0.7334
Epoch 105/150
 - 2s - loss: 0.1027 - acc: 0.9789 - val_loss: 1.6955 - val_acc: 0.7439
Epoch 106/150
 - 2s - loss: 0.1293 - acc: 0.9795 - val_loss: 1.5896 - val_acc: 0.7424
Epoch 107/150
 - 2s - loss: 0.1055 - acc: 0.9777 - val_loss: 1.5866 - val_acc: 0.7374
Epoch 108/150
 - 2s - loss: 0.0731 - acc: 0.9842 - val_loss: 1.6343 - val_acc: 0.7444
Epoch 109/150
 - 2s - loss: 0.0731 - acc: 0.9865 - val_loss: 1.6686 - val_acc: 0.7461
Epoch 110/150
 - 2s - loss: 0.0740 - acc: 0.9837 - val_loss: 1.6872 - val_acc: 0.7459
Epoch 111/150
```

```
 - 2s - loss: 0.0767 - acc: 0.9845 - val_loss: 1.7029 - val_acc: 0.7481
Epoch 112/150
 - 2s - loss: 0.1049 - acc: 0.9809 - val_loss: 1.6586 - val_acc: 0.7464
Epoch 113/150
 - 2s - loss: 0.0824 - acc: 0.9787 - val_loss: 1.6059 - val_acc: 0.7461
Epoch 114/150
 - 2s - loss: 0.0809 - acc: 0.9857 - val_loss: 1.6637 - val_acc: 0.7431
Epoch 115/150
 - 2s - loss: 0.1076 - acc: 0.9847 - val_loss: 1.4935 - val_acc: 0.7379
Epoch 116/150
 - 2s - loss: 0.0066 - acc: 0.9987 - val_loss: 1.7346 - val_acc: 0.7449
Epoch 117/150
 - 2s - loss: 0.1521 - acc: 0.9690 - val_loss: 1.7627 - val_acc: 0.7424
Epoch 118/150
 - 2s - loss: 0.0758 - acc: 0.9834 - val_loss: 1.7712 - val_acc: 0.7456
Epoch 119/150
 - 2s - loss: 0.1159 - acc: 0.9784 - val_loss: 1.7732 - val_acc: 0.7486
Epoch 120/150
 - 2s - loss: 0.0869 - acc: 0.9832 - val_loss: 1.7826 - val_acc: 0.7399
Epoch 121/150
 - 2s - loss: 0.0034 - acc: 0.9998 - val_loss: 2.7654 - val_acc: 0.7229
Epoch 122/150
 - 2s - loss: 0.1401 - acc: 0.9784 - val_loss: 1.8667 - val_acc: 0.7446
Epoch 123/150
 - 2s - loss: 0.1681 - acc: 0.9684 - val_loss: 1.8016 - val_acc: 0.7459
Epoch 124/150
 - 2s - loss: 0.0970 - acc: 0.9794 - val_loss: 1.9067 - val_acc: 0.7416
Epoch 125/150
 - 2s - loss: 0.0991 - acc: 0.9810 - val_loss: 1.8177 - val_acc: 0.7439
Epoch 126/150
 - 2s - loss: 0.0894 - acc: 0.9815 - val_loss: 1.7544 - val_acc: 0.7381
Epoch 127/150
 - 2s - loss: 0.0026 - acc: 0.9997 - val_loss: 2.0095 - val_acc: 0.7426
Epoch 128/150
 - 2s - loss: 0.1068 - acc: 0.9817 - val_loss: 1.9758 - val_acc: 0.7394
Epoch 129/150
 - 2s - loss: 0.1469 - acc: 0.9719 - val_loss: 1.7354 - val_acc: 0.7391
Epoch 130/150
 - 2s - loss: 0.0062 - acc: 0.9988 - val_loss: 1.9041 - val_acc: 0.7441
Epoch 131/150
 - 2s - loss: 0.1151 - acc: 0.9825 - val_loss: 1.9168 - val_acc: 0.7287
Epoch 132/150
 - 2s - loss: 0.0599 - acc: 0.9845 - val_loss: 1.9067 - val_acc: 0.7419
Epoch 133/150
 - 2s - loss: 0.1345 - acc: 0.9772 - val_loss: 1.8467 - val_acc: 0.7451
Epoch 134/150
 - 2s - loss: 0.0672 - acc: 0.9880 - val_loss: 1.9845 - val_acc: 0.7289
Epoch 135/150
```

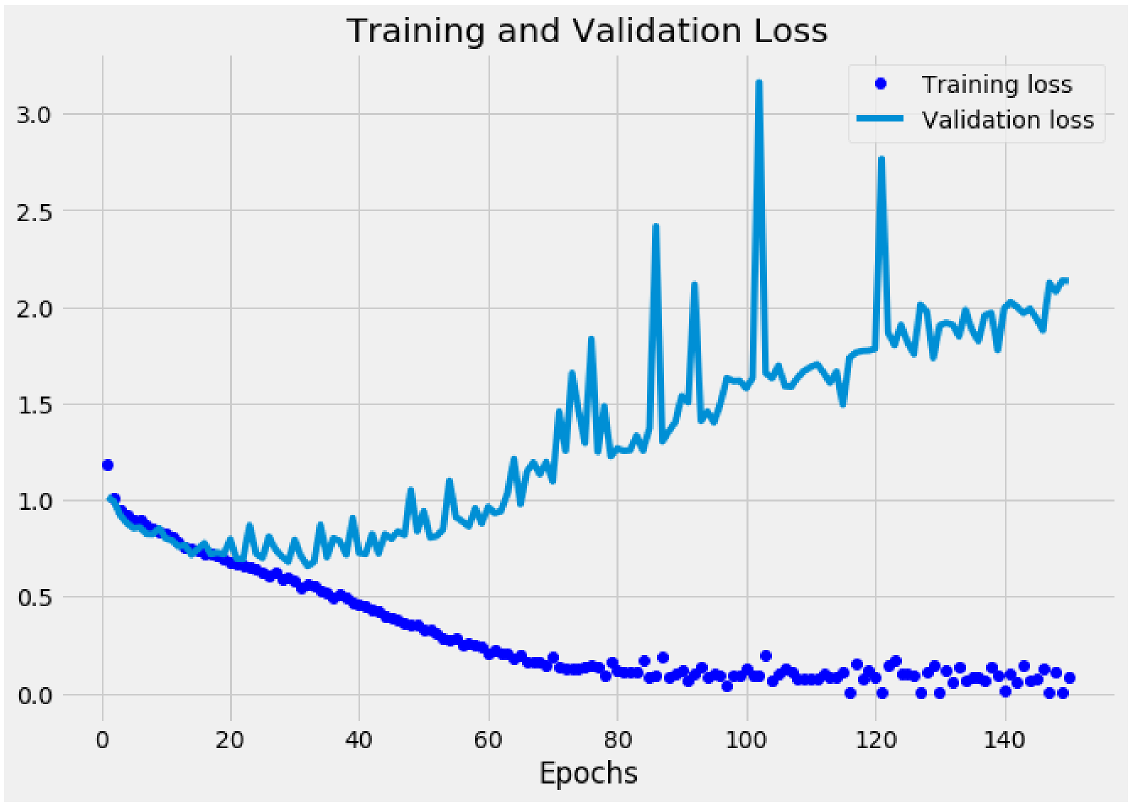
```
 - 2s - loss: 0.0812 - acc: 0.9819 - val_loss: 1.8802 - val_acc: 0.7461
Epoch 136/150
 - 2s - loss: 0.0829 - acc: 0.9845 - val_loss: 1.8222 - val_acc: 0.7456
Epoch 137/150
 - 2s - loss: 0.0676 - acc: 0.9837 - val_loss: 1.9559 - val_acc: 0.7441
Epoch 138/150
 - 2s - loss: 0.1361 - acc: 0.9785 - val_loss: 1.9677 - val_acc: 0.7332
Epoch 139/150
 - 2s - loss: 0.0936 - acc: 0.9837 - val_loss: 1.7754 - val_acc: 0.7122
Epoch 140/150
 - 2s - loss: 0.0153 - acc: 0.9955 - val_loss: 1.9922 - val_acc: 0.7409
Epoch 141/150
 - 2s - loss: 0.1004 - acc: 0.9814 - val_loss: 2.0247 - val_acc: 0.7461
Epoch 142/150
 - 2s - loss: 0.0528 - acc: 0.9875 - val_loss: 1.9988 - val_acc: 0.7429
Epoch 143/150
 - 2s - loss: 0.1473 - acc: 0.9749 - val_loss: 1.9665 - val_acc: 0.7481
Epoch 144/150
 - 2s - loss: 0.0652 - acc: 0.9850 - val_loss: 1.9898 - val_acc: 0.7421
Epoch 145/150
 - 2s - loss: 0.0785 - acc: 0.9829 - val_loss: 1.9389 - val_acc: 0.7436
Epoch 146/150
 - 2s - loss: 0.1248 - acc: 0.9790 - val_loss: 1.8780 - val_acc: 0.7309
Epoch 147/150
 - 2s - loss: 0.0037 - acc: 0.9990 - val_loss: 2.1234 - val_acc: 0.7449
Epoch 148/150
 - 2s - loss: 0.1072 - acc: 0.9789 - val_loss: 2.0778 - val_acc: 0.7329
Epoch 149/150
 - 2s - loss: 0.0030 - acc: 0.9995 - val_loss: 2.1351 - val_acc: 0.7361
Epoch 150/150
 - 2s - loss: 0.0872 - acc: 0.9835 - val_loss: 2.1339 - val_acc: 0.7419
```

[338]:
```
model.save('model_s.h5')
```

[339]:
```
fig = plt.figure(figsize=(10, 15))

accuracy = history.history['acc']
val_accuracy = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']

plt.subplot(2,1,1)
epochs = range(1, len(accuracy) + 1)
plt.plot(epochs, accuracy, 'bo', label='Training acc')
plt.plot(epochs, val_accuracy, label='Validation acc')
plt.title('Training and Validation Accuracy')
plt.xlabel('Epochs')
plt.legend()
fig1 = plt.figure()
fig1.savefig('train_val_acc_s.png')

fig = plt.figure(figsize=(10, 15))
plt.subplot(2,1,2)
plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, label='Validation loss')
plt.title('Training and Validation Loss')
plt.xlabel('Epochs')
plt.legend()
plt.figure()
plt.show();
fig.savefig('train_val_loss_s.png')
```



```
<Figure size 432x288 with 0 Axes>
```



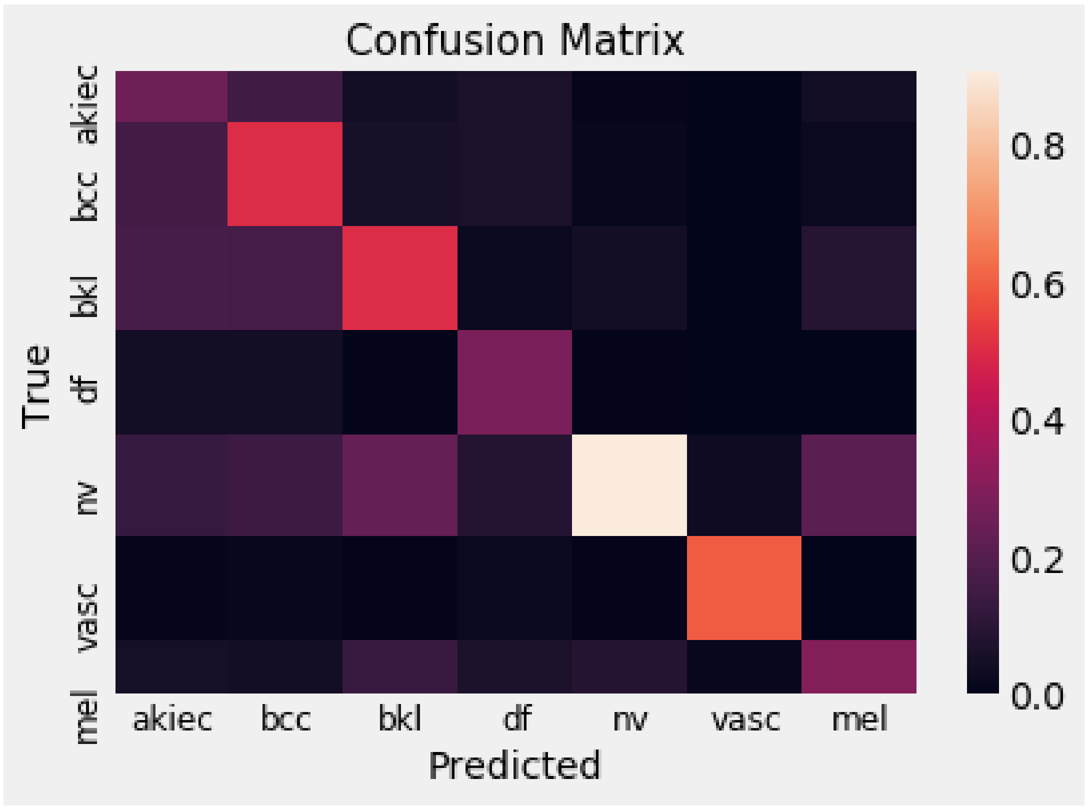
```
<Figure size 432x288 with 0 Axes>
```

[340]:
```
y_predict_s = model.predict(x_test_s)

y_predict_classes_s = np.argmax(y_predict_s,axis = 1)
y_true = np.argmax(y_test_s,axis = 1)
cm_1 = confusion_matrix(y_true,y_predict_classes_s)
cm_1
```

[340]:
```
array([[  33,   32,   20,    3,   24,    0,   19],
       [  21,  104,   25,    3,   42,    0,   11],
       [  22,   34,  222,    1,  120,    0,   40],
       [   6,    9,    3,   13,   14,    0,    1],
       [  17,   30,  103,    4, 2431,    2,   95],
       [   1,    3,    2,    1,   16,   34,    0],
       [   7,    9,   59,    3,  231,    1,  135]])
```

[341]:
```
ax= plt.subplot()
sns.heatmap(cm_1 / cm_1.astype(np.float).sum(axis=1), annot=False, ax=ax)

# labels, title and ticks
```

```
ax.set_xlabel('Predicted', size=14);
ax.set_ylabel('True', size=14);
ax.set_title('Confusion Matrix', size=15);
ax.xaxis.set_ticklabels(['akiec','bcc','bkl','df','nv','vasc','mel'], size=12);␣
 ↪\
ax.yaxis.set_ticklabels(['akiec','bcc','bkl','df','nv','vasc','mel'],size=12);
```



[342]:
```
precision, recall, fscore, _ = precision_recall_fscore_support(y_true,␣
 ↪y_predict_classes_s, average='weighted')
print('precision: {}'.format(precision))
print('recall: {}'.format(recall))
print('fscore: {}'.format(fscore))
```

```
precision: 0.7240795960440264
recall: 0.7418871692461309
fscore: 0.7298312223209095
```

[343]:
```
precision, recall, fscore, support = precision_recall_fscore_support(y_true,␣
 ↪y_predict_classes_s)
print('precision: {}'.format(precision))
print('recall: {}'.format(recall))
print('fscore: {}'.format(fscore))
```

```
precision: [0.30841121 0.47058824 0.51152074 0.46428571 0.84468381 0.91891892
 0.44850498]
recall: [0.2519084  0.50485437 0.50569476 0.2826087  0.90641312 0.59649123
 0.30337079]
fscore: [0.27731092 0.48711944 0.50859107 0.35135135 0.87446043 0.72340426
 0.36193029]
```

Accuracy are high but outputs are unstable and there is sign of overfitting. Deterioation is significant after epoch 20. If epoch = 20,

[344]:
```python
model_1a = Sequential()

model_1a.add(Conv2D(32, (3, 3), activation='relu', input_shape=(28,28,3)))
model_1a.add(MaxPool2D((2, 2)))
model_1a.add(Conv2D(64, (3, 3), activation='relu'))
model_1a.add(MaxPool2D((2, 2)))
model_1a.add(Conv2D(128, (3, 3), activation='relu'))
model_1a.add(MaxPool2D((2, 2)))

model_1a.add(Flatten())

model_1a.add(Dense(512, activation='relu'))
model_1a.add(Dense(7))
model_1a.add(Activation(tf.nn.softmax))


#model = Model(inputs, outputs)
print(model_1a.summary())
```

```
Model: "sequential_16"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_53 (Conv2D)           (None, 26, 26, 32)        896
_________________________________________________________________
max_pooling2d_43 (MaxPooling (None, 13, 13, 32)        0
_________________________________________________________________
conv2d_54 (Conv2D)           (None, 11, 11, 64)        18496
_________________________________________________________________
max_pooling2d_44 (MaxPooling (None, 5, 5, 64)          0
_________________________________________________________________
conv2d_55 (Conv2D)           (None, 3, 3, 128)         73856
_________________________________________________________________
max_pooling2d_45 (MaxPooling (None, 1, 1, 128)         0
_________________________________________________________________
flatten_16 (Flatten)         (None, 128)               0
_________________________________________________________________
dense_34 (Dense)             (None, 512)               66048
```

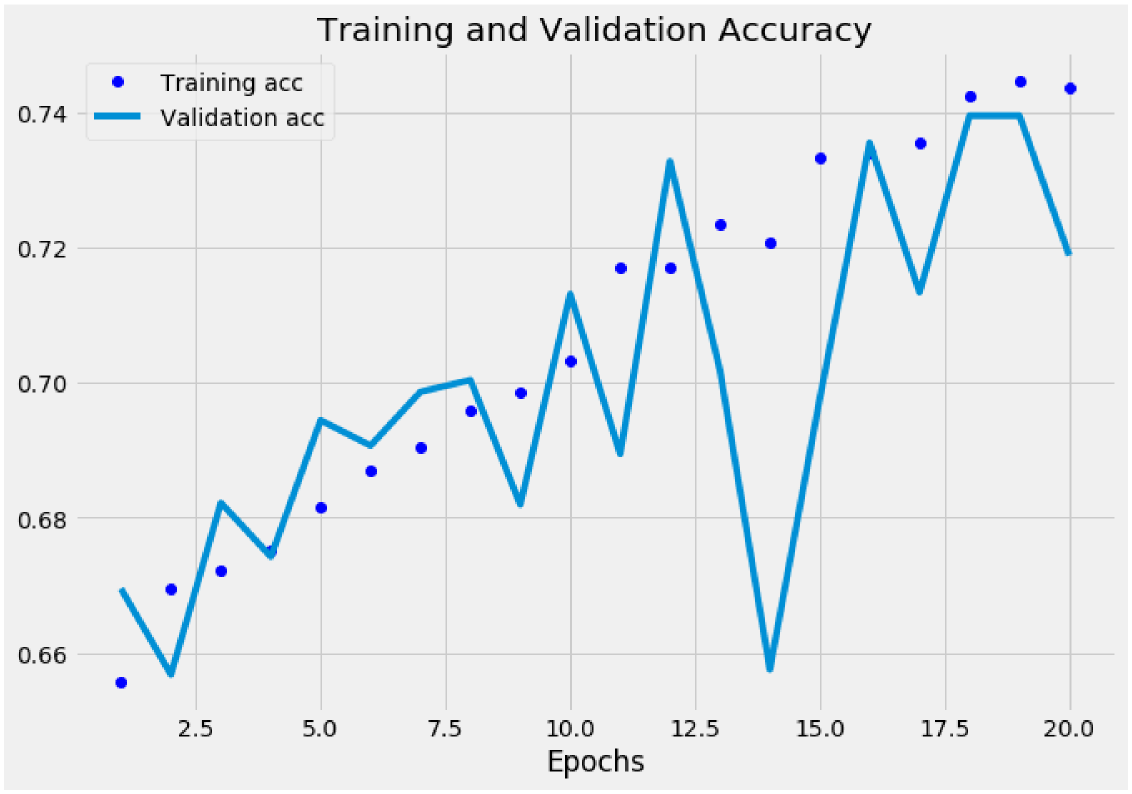
```
_________________________________________________________________
dense_35 (Dense)             (None, 7)                 3591
_________________________________________________________________
activation_9 (Activation)    (None, 7)                 0
=================================================================
Total params: 162,887
Trainable params: 162,887
Non-trainable params: 0
_________________________________________________________________
None
```

[345]:
```
model_1a.compile(loss='categorical_crossentropy',
optimizer='rmsprop',
metrics=['acc'])

history_1a = model_1a.fit(
    x_train_s, y_train_s,
    epochs=20,
    batch_size = 128,
    validation_data=(x_test_s, y_test_s),
    verbose=2)
```

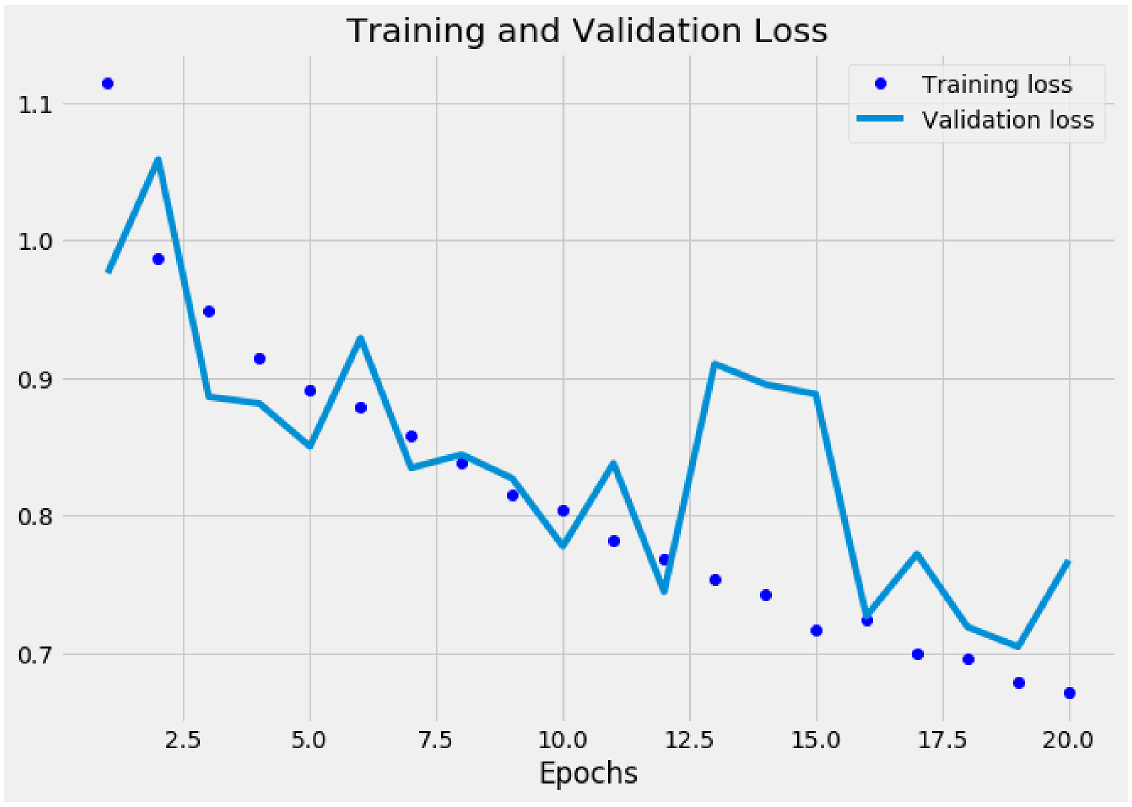
```
Train on 6008 samples, validate on 4006 samples
Epoch 1/20
 - 2s - loss: 1.1145 - acc: 0.6556 - val_loss: 0.9759 - val_acc: 0.6695
Epoch 2/20
 - 2s - loss: 0.9862 - acc: 0.6696 - val_loss: 1.0585 - val_acc: 0.6568
Epoch 3/20
 - 2s - loss: 0.9483 - acc: 0.6721 - val_loss: 0.8862 - val_acc: 0.6822
Epoch 4/20
 - 2s - loss: 0.9146 - acc: 0.6751 - val_loss: 0.8814 - val_acc: 0.6742
Epoch 5/20
 - 2s - loss: 0.8909 - acc: 0.6816 - val_loss: 0.8502 - val_acc: 0.6945
Epoch 6/20
 - 2s - loss: 0.8786 - acc: 0.6871 - val_loss: 0.9289 - val_acc: 0.6907
Epoch 7/20
 - 2s - loss: 0.8575 - acc: 0.6906 - val_loss: 0.8346 - val_acc: 0.6987
Epoch 8/20
 - 2s - loss: 0.8386 - acc: 0.6959 - val_loss: 0.8443 - val_acc: 0.7004
Epoch 9/20
 - 2s - loss: 0.8156 - acc: 0.6986 - val_loss: 0.8270 - val_acc: 0.6820
Epoch 10/20
 - 2s - loss: 0.8037 - acc: 0.7034 - val_loss: 0.7777 - val_acc: 0.7132
Epoch 11/20
 - 2s - loss: 0.7820 - acc: 0.7172 - val_loss: 0.8380 - val_acc: 0.6895
Epoch 12/20
 - 2s - loss: 0.7683 - acc: 0.7172 - val_loss: 0.7448 - val_acc: 0.7329
Epoch 13/20
```

```
 - 2s - loss: 0.7543 - acc: 0.7235 - val_loss: 0.9102 - val_acc: 0.7019
Epoch 14/20
 - 2s - loss: 0.7429 - acc: 0.7209 - val_loss: 0.8953 - val_acc: 0.6575
Epoch 15/20
 - 2s - loss: 0.7175 - acc: 0.7335 - val_loss: 0.8882 - val_acc: 0.6977
Epoch 16/20
 - 2s - loss: 0.7244 - acc: 0.7342 - val_loss: 0.7271 - val_acc: 0.7356
Epoch 17/20
 - 2s - loss: 0.7003 - acc: 0.7357 - val_loss: 0.7722 - val_acc: 0.7134
Epoch 18/20
 - 2s - loss: 0.6966 - acc: 0.7425 - val_loss: 0.7192 - val_acc: 0.7396
Epoch 19/20
 - 2s - loss: 0.6793 - acc: 0.7448 - val_loss: 0.7048 - val_acc: 0.7396
Epoch 20/20
 - 2s - loss: 0.6715 - acc: 0.7437 - val_loss: 0.7676 - val_acc: 0.7189
```

[346]:
```
fig = plt.figure(figsize=(10, 15))

accuracy = history_1a.history['acc']
val_accuracy = history_1a.history['val_acc']
loss = history_1a.history['loss']
val_loss = history_1a.history['val_loss']

plt.subplot(2,1,1)
epochs = range(1, len(accuracy) + 1)
plt.plot(epochs, accuracy, 'bo', label='Training acc')
plt.plot(epochs, val_accuracy, label='Validation acc')
plt.title('Training and Validation Accuracy')
plt.xlabel('Epochs')
plt.legend()
fig1 = plt.figure()
fig1.savefig('train_val_acc_s.png')

fig = plt.figure(figsize=(10, 15))
plt.subplot(2,1,2)
plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, label='Validation loss')
plt.title('Training and Validation Loss')
plt.xlabel('Epochs')
plt.legend()
plt.figure()
plt.show();
fig.savefig('train_val_loss_s.png')
```


Training and Validation Accuracy
Training acc
Validation acc
0.74
0.72
0.70
0.68
0.66
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Epochs

```
<Figure size 432x288 with 0 Axes>
```



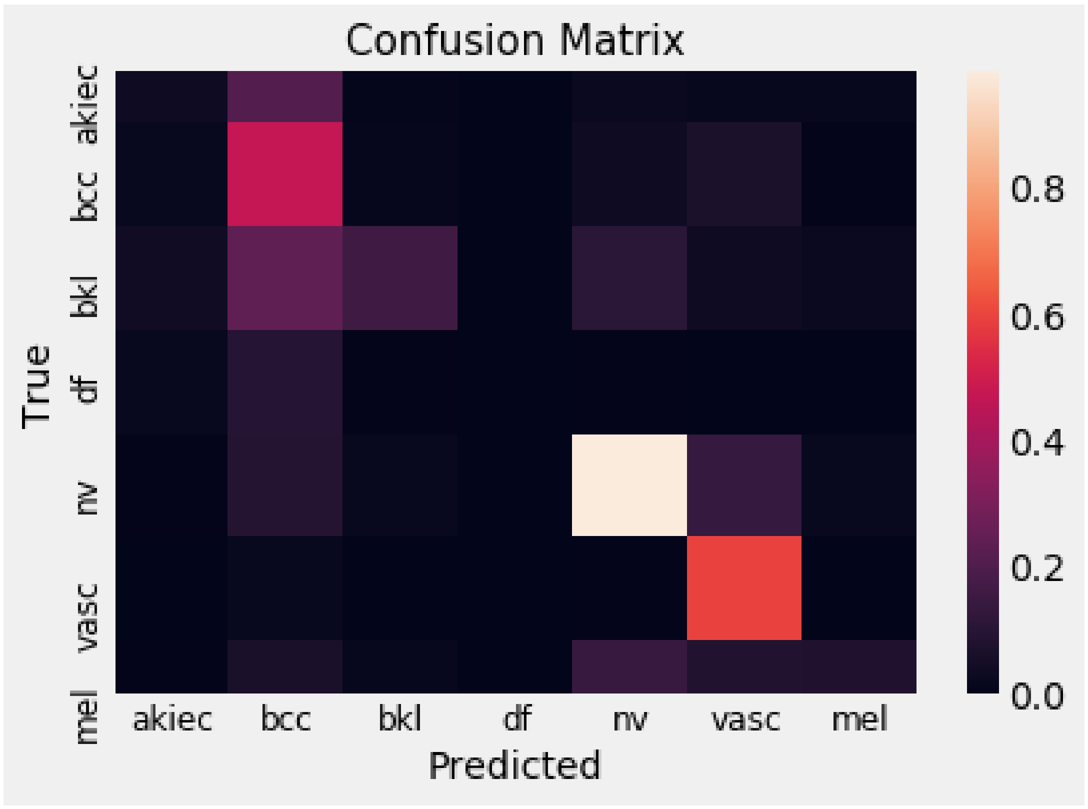
```
<Figure size 432x288 with 0 Axes>
```

[347]:
```
y_predict_s = model_1a.predict(x_test_s)

y_predict_classes_s = np.argmax(y_predict_s,axis = 1)
y_true = np.argmax(y_test_s,axis = 1)
cm_2 = confusion_matrix(y_true,y_predict_classes_s)
cm_2
```

[347]:
```
array([[   5,   44,    5,    0,   68,    1,    8],
       [   3,   98,    6,    0,   93,    4,    2],
       [   6,   50,   72,    0,  297,    2,   12],
       [   3,   20,    3,    0,   19,    0,    1],
       [   1,   19,   10,    0, 2635,    8,    9],
       [   0,    4,    0,    0,   18,   34,    1],
       [   1,   14,    8,    0,  381,    5,   36]])
```

[348]:
```
ax= plt.subplot()
sns.heatmap(cm_2 / cm_2.astype(np.float).sum(axis=1), annot=False, ax=ax)

# labels, title and ticks
```

```
ax.set_xlabel('Predicted', size=14);
ax.set_ylabel('True', size=14);
ax.set_title('Confusion Matrix', size=15);
ax.xaxis.set_ticklabels(['akiec','bcc','bkl','df','nv','vasc','mel'], size=12);␣
↪\
ax.yaxis.set_ticklabels(['akiec','bcc','bkl','df','nv','vasc','mel'], size=12);
```



[349]:
```
precision, recall, fscore, _ = precision_recall_fscore_support(y_true,␣
↪y_predict_classes_s, average='weighted')
print('precision: {}'.format(precision))
print('recall: {}'.format(recall))
print('fscore: {}'.format(fscore))
```

```
precision: 0.6740820436644339
recall: 0.7189216175736396
fscore: 0.6473844227988411
```

```
/opt/anaconda3/lib/python3.7/site-
packages/sklearn/metrics/_classification.py:1272: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

[350]:
```
precision, recall, fscore, _ = precision_recall_fscore_support(y_true,␣
 ↪y_predict_classes_s)
print('precision: {}'.format(precision))
print('recall: {}'.format(recall))
print('fscore: {}'.format(fscore))
```

```
precision: [0.26315789 0.3935743  0.69230769 0.         0.75049843 0.62962963
 0.52173913]
recall: [0.03816794 0.47572816 0.16400911 0.         0.98247576 0.59649123
 0.08089888]
fscore: [0.06666667 0.43076923 0.26519337 0.         0.85096076 0.61261261
 0.14007782]
```

[351]:
```
# tweaking the model by adding dropout & hidden layer

x_train_st, x_test_st, y_train_st, y_test_st = train_test_split(x, y,␣
 ↪test_size=0.5)
print(x_train_st.shape)
print(y_train_st.shape)
print(x_test_st.shape)
print(y_test_st.shape)
```

```
(5007, 28, 28, 3)
(5007,)
(5007, 28, 28, 3)
(5007,)
```

[352]:
```
model_2 = Sequential()

model_2.add(Conv2D(32, (3, 3), padding='same', activation='relu',␣
 ↪input_shape=(28,28,3)))
model_2.add(MaxPool2D((2, 2)))

model_2.add(Conv2D(64, (3, 3), activation='relu'))
model_2.add(MaxPool2D((2, 2)))

model_2.add(Conv2D(128, (3, 3), activation='relu'))
model_2.add(Conv2D(128, (3, 3), activation='relu'))
model_2.add(MaxPool2D((2, 2)))

model_2.add(Flatten())

model_2.add(Dense(250, activation='relu'))
model_2.add(Dropout(0.5))
model_2.add(Dense(250, activation='relu'))
model_2.add(Dropout(0.5))
model_2.add(Dense(7, activation='softmax'))
```

```
#model.add(Activation(tf.nn.softmax))


#model = Model(inputs, outputs)
print(model_2.summary())
```

```
Model: "sequential_17"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_56 (Conv2D)           (None, 28, 28, 32)        896
_________________________________________________________________
max_pooling2d_46 (MaxPooling (None, 14, 14, 32)        0
_________________________________________________________________
conv2d_57 (Conv2D)           (None, 12, 12, 64)        18496
_________________________________________________________________
max_pooling2d_47 (MaxPooling (None, 6, 6, 64)          0
_________________________________________________________________
conv2d_58 (Conv2D)           (None, 4, 4, 128)         73856
_________________________________________________________________
conv2d_59 (Conv2D)           (None, 2, 2, 128)         147584
_________________________________________________________________
max_pooling2d_48 (MaxPooling (None, 1, 1, 128)         0
_________________________________________________________________
flatten_17 (Flatten)         (None, 128)               0
_________________________________________________________________
dense_36 (Dense)             (None, 250)               32250
_________________________________________________________________
dropout_16 (Dropout)         (None, 250)               0
_________________________________________________________________
dense_37 (Dense)             (None, 250)               62750
_________________________________________________________________
dropout_17 (Dropout)         (None, 250)               0
_________________________________________________________________
dense_38 (Dense)             (None, 7)                 1757
=================================================================
Total params: 337,589
Trainable params: 337,589
Non-trainable params: 0
_________________________________________________________________
None
```

[353]:
```
model_2.compile('rmsprop', loss='categorical_crossentropy', metrics=['acc'])
history_2 = model_2.fit(
    x, to_categorical(y),
    epochs=150, batch_size=200,
    validation_split=0.1,
    verbose=1,
    callbacks=[ModelCheckpoint('model_2.h5', save_best_only=True)]
)
```

```
Train on 9012 samples, validate on 1002 samples
Epoch 1/150
9012/9012 [==============================] - 5s 521us/step - loss: 1.2050 - acc:
0.6580 - val_loss: 1.8729 - val_acc: 0.6697
Epoch 2/150
9012/9012 [==============================] - 4s 469us/step - loss: 0.9746 - acc:
0.6692 - val_loss: 2.4536 - val_acc: 0.6697
Epoch 3/150
9012/9012 [==============================] - 4s 477us/step - loss: 0.9095 - acc:
0.6719 - val_loss: 2.0925 - val_acc: 0.4261
Epoch 4/150
9012/9012 [==============================] - 4s 473us/step - loss: 0.8586 - acc:
0.6835 - val_loss: 4.2488 - val_acc: 0.0030
Epoch 5/150
9012/9012 [==============================] - 4s 477us/step - loss: 0.8520 - acc:
0.6825 - val_loss: 4.3600 - val_acc: 0.1507
Epoch 6/150
9012/9012 [==============================] - 4s 476us/step - loss: 0.8216 - acc:
0.6902 - val_loss: 3.4265 - val_acc: 0.6657
Epoch 7/150
9012/9012 [==============================] - 4s 469us/step - loss: 0.8046 - acc:
0.6911 - val_loss: 2.6566 - val_acc: 0.6697
Epoch 8/150
9012/9012 [==============================] - 4s 468us/step - loss: 0.7736 - acc:
0.7006 - val_loss: 2.4362 - val_acc: 0.6297
Epoch 9/150
9012/9012 [==============================] - 4s 472us/step - loss: 0.7566 - acc:
0.7099 - val_loss: 3.0131 - val_acc: 0.5309
Epoch 10/150
9012/9012 [==============================] - 4s 472us/step - loss: 0.7299 - acc:
0.7163 - val_loss: 4.8020 - val_acc: 0.5898
Epoch 11/150
9012/9012 [==============================] - 4s 475us/step - loss: 0.7033 - acc:
0.7309 - val_loss: 6.8513 - val_acc: 0.4242
Epoch 12/150
9012/9012 [==============================] - 4s 475us/step - loss: 0.6889 - acc:
0.7362 - val_loss: 5.0531 - val_acc: 0.5060
Epoch 13/150
9012/9012 [==============================] - 4s 473us/step - loss: 0.6668 - acc:
0.7403 - val_loss: 6.6399 - val_acc: 0.5569
Epoch 14/150
9012/9012 [==============================] - 4s 473us/step - loss: 0.6596 - acc:
0.7487 - val_loss: 5.9668 - val_acc: 0.6238
```

```
Epoch 15/150
9012/9012 [==============================] - 4s 473us/step - loss: 0.6455 - acc:
0.7494 - val_loss: 5.1072 - val_acc: 0.5419
Epoch 16/150
9012/9012 [==============================] - 4s 476us/step - loss: 0.6314 - acc:
0.7602 - val_loss: 4.8302 - val_acc: 0.5878
Epoch 17/150
9012/9012 [==============================] - 4s 471us/step - loss: 0.6202 - acc:
0.7673 - val_loss: 5.5695 - val_acc: 0.5868
Epoch 18/150
9012/9012 [==============================] - 4s 473us/step - loss: 0.5996 - acc:
0.7698 - val_loss: 6.0975 - val_acc: 0.2255
Epoch 19/150
9012/9012 [==============================] - 4s 474us/step - loss: 0.5884 - acc:
0.7747 - val_loss: 5.5011 - val_acc: 0.5739
Epoch 20/150
9012/9012 [==============================] - 4s 474us/step - loss: 0.5799 - acc:
0.7773 - val_loss: 8.9236 - val_acc: 0.2575
Epoch 21/150
9012/9012 [==============================] - 4s 474us/step - loss: 0.5541 - acc:
0.7851 - val_loss: 7.1658 - val_acc: 0.3593
Epoch 22/150
9012/9012 [==============================] - 4s 473us/step - loss: 0.5439 - acc:
0.7947 - val_loss: 6.6204 - val_acc: 0.5599
Epoch 23/150
9012/9012 [==============================] - 4s 479us/step - loss: 0.5296 - acc:
0.8018 - val_loss: 6.3237 - val_acc: 0.4820
Epoch 24/150
9012/9012 [==============================] - 4s 477us/step - loss: 0.5211 - acc:
0.8054 - val_loss: 8.2043 - val_acc: 0.6427
Epoch 25/150
9012/9012 [==============================] - 4s 480us/step - loss: 0.5059 - acc:
0.8083 - val_loss: 7.2889 - val_acc: 0.5180
Epoch 26/150
9012/9012 [==============================] - 4s 480us/step - loss: 0.4831 - acc:
0.8179 - val_loss: 7.8504 - val_acc: 0.5958
Epoch 27/150
9012/9012 [==============================] - 4s 480us/step - loss: 0.4891 - acc:
0.8151 - val_loss: 8.0635 - val_acc: 0.2884
Epoch 28/150
9012/9012 [==============================] - 4s 477us/step - loss: 0.4446 - acc:
0.8328 - val_loss: 9.4611 - val_acc: 0.5349
Epoch 29/150
9012/9012 [==============================] - 4s 479us/step - loss: 0.4528 - acc:
0.8309 - val_loss: 10.4287 - val_acc: 0.4062
Epoch 30/150
9012/9012 [==============================] - 4s 481us/step - loss: 0.4250 - acc:
0.8408 - val_loss: 9.6446 - val_acc: 0.4920
```

```
Epoch 31/150
9012/9012 [==============================] - 4s 480us/step - loss: 0.4085 - acc:
0.8470 - val_loss: 11.4820 - val_acc: 0.6327
Epoch 32/150
9012/9012 [==============================] - 4s 479us/step - loss: 0.4225 - acc:
0.8474 - val_loss: 10.6986 - val_acc: 0.4780
Epoch 33/150
9012/9012 [==============================] - 4s 481us/step - loss: 0.3753 - acc:
0.8593 - val_loss: 10.1567 - val_acc: 0.6038
Epoch 34/150
9012/9012 [==============================] - 4s 481us/step - loss: 0.3872 - acc:
0.8587 - val_loss: 10.6195 - val_acc: 0.4641
Epoch 35/150
9012/9012 [==============================] - 4s 479us/step - loss: 0.3481 - acc:
0.8723 - val_loss: 9.2143 - val_acc: 0.5000
Epoch 36/150
9012/9012 [==============================] - 4s 478us/step - loss: 0.3485 - acc:
0.8727 - val_loss: 10.9094 - val_acc: 0.4840
Epoch 37/150
9012/9012 [==============================] - 4s 476us/step - loss: 0.3370 - acc:
0.8768 - val_loss: 8.7009 - val_acc: 0.5190
Epoch 38/150
9012/9012 [==============================] - 4s 477us/step - loss: 0.3037 - acc:
0.8904 - val_loss: 10.7572 - val_acc: 0.4581
Epoch 39/150
9012/9012 [==============================] - 4s 476us/step - loss: 0.3090 - acc:
0.8899 - val_loss: 9.9034 - val_acc: 0.5220
Epoch 40/150
9012/9012 [==============================] - 4s 477us/step - loss: 0.2844 - acc:
0.8949 - val_loss: 10.4089 - val_acc: 0.4451
Epoch 41/150
9012/9012 [==============================] - 4s 479us/step - loss: 0.2711 - acc:
0.8997 - val_loss: 11.2138 - val_acc: 0.4721
Epoch 42/150
9012/9012 [==============================] - 4s 476us/step - loss: 0.2650 - acc:
0.9037 - val_loss: 18.5124 - val_acc: 0.3703
Epoch 43/150
9012/9012 [==============================] - 4s 479us/step - loss: 0.2615 - acc:
0.9107 - val_loss: 10.6417 - val_acc: 0.3413
Epoch 44/150
9012/9012 [==============================] - 4s 480us/step - loss: 0.2380 - acc:
0.9161 - val_loss: 11.2504 - val_acc: 0.2255
Epoch 45/150
9012/9012 [==============================] - 4s 477us/step - loss: 0.2436 - acc:
0.9144 - val_loss: 12.8530 - val_acc: 0.5409
Epoch 46/150
9012/9012 [==============================] - 4s 477us/step - loss: 0.2254 - acc:
0.9208 - val_loss: 11.2064 - val_acc: 0.5429
```

```
Epoch 47/150
9012/9012 [==============================] - 4s 481us/step - loss: 0.2091 - acc:
0.9284 - val_loss: 14.4637 - val_acc: 0.4631
Epoch 48/150
9012/9012 [==============================] - 4s 476us/step - loss: 0.2226 - acc:
0.9244 - val_loss: 15.4702 - val_acc: 0.5599
Epoch 49/150
9012/9012 [==============================] - 4s 478us/step - loss: 0.2136 - acc:
0.9288 - val_loss: 11.9611 - val_acc: 0.4032
Epoch 50/150
9012/9012 [==============================] - 4s 477us/step - loss: 0.1875 - acc:
0.9343 - val_loss: 12.2291 - val_acc: 0.4651
Epoch 51/150
9012/9012 [==============================] - 4s 478us/step - loss: 0.1794 - acc:
0.9414 - val_loss: 10.3501 - val_acc: 0.5868
Epoch 52/150
9012/9012 [==============================] - 4s 477us/step - loss: 0.1674 - acc:
0.9445 - val_loss: 12.6848 - val_acc: 0.5599
Epoch 53/150
9012/9012 [==============================] - 4s 473us/step - loss: 0.1681 - acc:
0.9456 - val_loss: 11.3506 - val_acc: 0.4631
Epoch 54/150
9012/9012 [==============================] - 4s 474us/step - loss: 0.1954 - acc:
0.9352 - val_loss: 12.9832 - val_acc: 0.4840
Epoch 55/150
9012/9012 [==============================] - 4s 468us/step - loss: 0.1409 - acc:
0.9503 - val_loss: 13.2729 - val_acc: 0.5279
Epoch 56/150
9012/9012 [==============================] - 4s 467us/step - loss: 0.1741 - acc:
0.9467 - val_loss: 13.5476 - val_acc: 0.4331
Epoch 57/150
9012/9012 [==============================] - 4s 466us/step - loss: 0.1550 - acc:
0.9500 - val_loss: 16.3623 - val_acc: 0.4910
Epoch 58/150
9012/9012 [==============================] - 4s 469us/step - loss: 0.1242 - acc:
0.9558 - val_loss: 13.7388 - val_acc: 0.4561
Epoch 59/150
9012/9012 [==============================] - 4s 467us/step - loss: 0.1479 - acc:
0.9512 - val_loss: 12.8136 - val_acc: 0.5230
Epoch 60/150
9012/9012 [==============================] - 4s 473us/step - loss: 0.1405 - acc:
0.9566 - val_loss: 14.7390 - val_acc: 0.4661
Epoch 61/150
9012/9012 [==============================] - 4s 473us/step - loss: 0.1242 - acc:
0.9624 - val_loss: 14.7703 - val_acc: 0.4970
Epoch 62/150
9012/9012 [==============================] - 4s 472us/step - loss: 0.1333 - acc:
0.9581 - val_loss: 15.8578 - val_acc: 0.2465
```

```
Epoch 63/150
9012/9012 [==============================] - 4s 471us/step - loss: 0.1028 - acc:
0.9668 - val_loss: 18.4788 - val_acc: 0.4541
Epoch 64/150
9012/9012 [==============================] - 4s 471us/step - loss: 0.1198 - acc:
0.9617 - val_loss: 16.0740 - val_acc: 0.5269
Epoch 65/150
9012/9012 [==============================] - 4s 475us/step - loss: 0.1310 - acc:
0.9585 - val_loss: 16.2049 - val_acc: 0.3273
Epoch 66/150
9012/9012 [==============================] - 4s 471us/step - loss: 0.1337 - acc:
0.9645 - val_loss: 17.2224 - val_acc: 0.4521
Epoch 67/150
9012/9012 [==============================] - 4s 470us/step - loss: 0.1135 - acc:
0.9683 - val_loss: 11.8374 - val_acc: 0.4541
Epoch 68/150
9012/9012 [==============================] - 4s 471us/step - loss: 0.0882 - acc:
0.9714 - val_loss: 19.0965 - val_acc: 0.5170
Epoch 69/150
9012/9012 [==============================] - 4s 473us/step - loss: 0.0844 - acc:
0.9743 - val_loss: 13.8019 - val_acc: 0.2784
Epoch 70/150
9012/9012 [==============================] - 4s 473us/step - loss: 0.1106 - acc:
0.9640 - val_loss: 18.5437 - val_acc: 0.4571
Epoch 71/150
9012/9012 [==============================] - 4s 469us/step - loss: 0.0767 - acc:
0.9784 - val_loss: 18.5824 - val_acc: 0.4331
Epoch 72/150
9012/9012 [==============================] - 4s 473us/step - loss: 0.1178 - acc:
0.9686 - val_loss: 18.4154 - val_acc: 0.4501
Epoch 73/150
9012/9012 [==============================] - 4s 471us/step - loss: 0.0979 - acc:
0.9717 - val_loss: 17.2943 - val_acc: 0.4940
Epoch 74/150
9012/9012 [==============================] - 4s 469us/step - loss: 0.0811 - acc:
0.9739 - val_loss: 18.1769 - val_acc: 0.5050
Epoch 75/150
9012/9012 [==============================] - 4s 470us/step - loss: 0.1028 - acc:
0.9697 - val_loss: 16.7915 - val_acc: 0.5070
Epoch 76/150
9012/9012 [==============================] - 4s 471us/step - loss: 0.1078 - acc:
0.9703 - val_loss: 17.6290 - val_acc: 0.5319
Epoch 77/150
9012/9012 [==============================] - 4s 471us/step - loss: 0.1161 - acc:
0.9682 - val_loss: 20.8487 - val_acc: 0.5050
Epoch 78/150
9012/9012 [==============================] - 4s 472us/step - loss: 0.0594 - acc:
0.9799 - val_loss: 21.5849 - val_acc: 0.5220
```

```
Epoch 79/150
9012/9012 [==============================] - 4s 475us/step - loss: 0.1344 - acc:
0.9657 - val_loss: 18.2173 - val_acc: 0.4760
Epoch 80/150
9012/9012 [==============================] - 4s 472us/step - loss: 0.0895 - acc:
0.9767 - val_loss: 17.2069 - val_acc: 0.4601
Epoch 81/150
9012/9012 [==============================] - 4s 467us/step - loss: 0.0799 - acc:
0.9765 - val_loss: 15.9536 - val_acc: 0.5419
Epoch 82/150
9012/9012 [==============================] - 4s 471us/step - loss: 0.0820 - acc:
0.9775 - val_loss: 21.4090 - val_acc: 0.4810
Epoch 83/150
9012/9012 [==============================] - 4s 468us/step - loss: 0.0769 - acc:
0.9780 - val_loss: 23.4868 - val_acc: 0.4172
Epoch 84/150
9012/9012 [==============================] - 4s 473us/step - loss: 0.1060 - acc:
0.9714 - val_loss: 17.4049 - val_acc: 0.4072
Epoch 85/150
9012/9012 [==============================] - 4s 474us/step - loss: 0.0600 - acc:
0.9819 - val_loss: 17.1819 - val_acc: 0.3792
Epoch 86/150
9012/9012 [==============================] - 4s 472us/step - loss: 0.1220 - acc:
0.9676 - val_loss: 17.2953 - val_acc: 0.4810
Epoch 87/150
9012/9012 [==============================] - 4s 471us/step - loss: 0.0656 - acc:
0.9804 - val_loss: 19.0177 - val_acc: 0.4671
Epoch 88/150
9012/9012 [==============================] - 4s 472us/step - loss: 0.0778 - acc:
0.9784 - val_loss: 22.6877 - val_acc: 0.4381
Epoch 89/150
9012/9012 [==============================] - 4s 472us/step - loss: 0.0796 - acc:
0.9763 - val_loss: 20.9764 - val_acc: 0.4721
Epoch 90/150
9012/9012 [==============================] - 4s 472us/step - loss: 0.0818 - acc:
0.9763 - val_loss: 16.9085 - val_acc: 0.5419
Epoch 91/150
9012/9012 [==============================] - 4s 471us/step - loss: 0.0549 - acc:
0.9840 - val_loss: 26.5570 - val_acc: 0.3912
Epoch 92/150
9012/9012 [==============================] - 4s 490us/step - loss: 0.1184 - acc:
0.9719 - val_loss: 18.9651 - val_acc: 0.4351
Epoch 93/150
9012/9012 [==============================] - 4s 498us/step - loss: 0.0774 - acc:
0.9778 - val_loss: 25.3021 - val_acc: 0.2904
Epoch 94/150
9012/9012 [==============================] - 4s 479us/step - loss: 0.1209 - acc:
0.9726 - val_loss: 18.9178 - val_acc: 0.4741
```

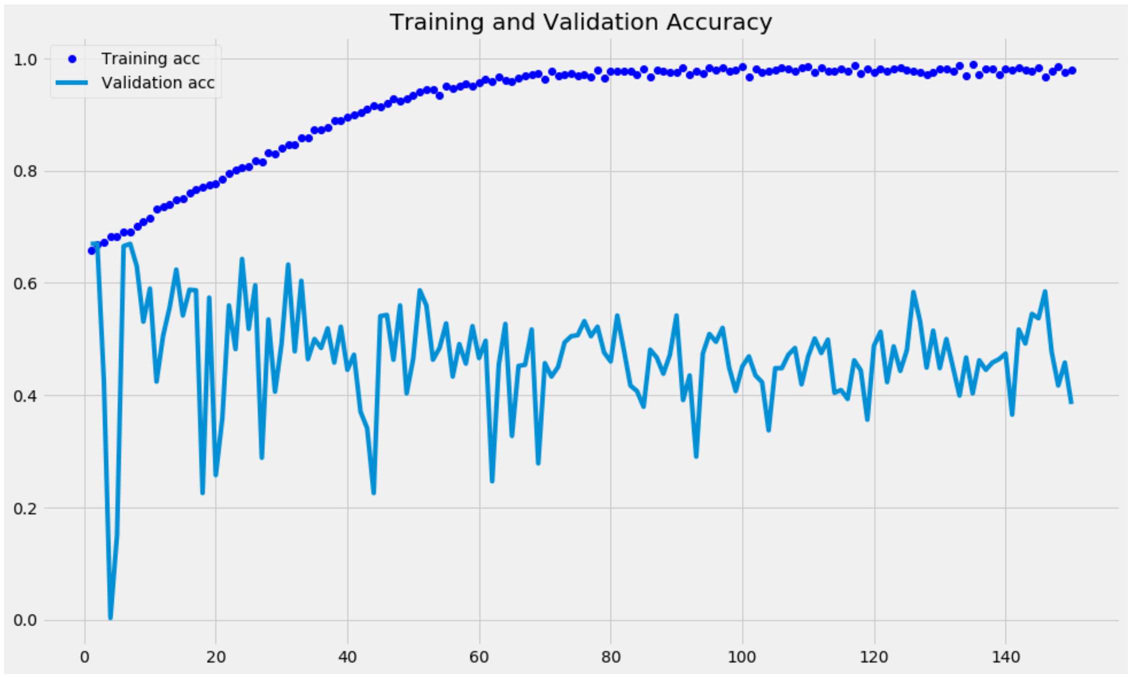
```
Epoch 95/150
9012/9012 [==============================] - 4s 480us/step - loss: 0.0580 - acc:
0.9834 - val_loss: 23.9776 - val_acc: 0.5090
Epoch 96/150
9012/9012 [==============================] - 4s 485us/step - loss: 0.0685 - acc:
0.9796 - val_loss: 16.2926 - val_acc: 0.4950
Epoch 97/150
9012/9012 [==============================] - 4s 481us/step - loss: 0.0573 - acc:
0.9836 - val_loss: 19.4234 - val_acc: 0.5200
Epoch 98/150
9012/9012 [==============================] - 4s 486us/step - loss: 0.0676 - acc:
0.9768 - val_loss: 19.5940 - val_acc: 0.4491
Epoch 99/150
9012/9012 [==============================] - 4s 487us/step - loss: 0.0757 - acc:
0.9792 - val_loss: 20.1714 - val_acc: 0.4072
Epoch 100/150
9012/9012 [==============================] - 4s 497us/step - loss: 0.0513 - acc:
0.9860 - val_loss: 24.6550 - val_acc: 0.4511
Epoch 101/150
9012/9012 [==============================] - 4s 478us/step - loss: 0.1198 - acc:
0.9676 - val_loss: 18.0573 - val_acc: 0.4691
Epoch 102/150
9012/9012 [==============================] - 4s 478us/step - loss: 0.0747 - acc:
0.9806 - val_loss: 23.7456 - val_acc: 0.4351
Epoch 103/150
9012/9012 [==============================] - 4s 476us/step - loss: 0.0835 - acc:
0.9754 - val_loss: 19.7156 - val_acc: 0.4232
Epoch 104/150
9012/9012 [==============================] - 4s 477us/step - loss: 0.1193 - acc:
0.9770 - val_loss: 25.1311 - val_acc: 0.3373
Epoch 105/150
9012/9012 [==============================] - 4s 477us/step - loss: 0.0801 - acc:
0.9795 - val_loss: 20.4324 - val_acc: 0.4481
Epoch 106/150
9012/9012 [==============================] - 4s 479us/step - loss: 0.0517 - acc:
0.9840 - val_loss: 21.3650 - val_acc: 0.4481
Epoch 107/150
9012/9012 [==============================] - 4s 479us/step - loss: 0.0789 - acc:
0.9807 - val_loss: 20.8147 - val_acc: 0.4721
Epoch 108/150
9012/9012 [==============================] - 4s 477us/step - loss: 0.0861 - acc:
0.9767 - val_loss: 20.4517 - val_acc: 0.4840
Epoch 109/150
9012/9012 [==============================] - 5s 515us/step - loss: 0.0528 - acc:
0.9829 - val_loss: 22.0932 - val_acc: 0.4192
Epoch 110/150
9012/9012 [==============================] - 5s 505us/step - loss: 0.0494 - acc:
0.9854 - val_loss: 30.3674 - val_acc: 0.4691
```

```
Epoch 111/150
9012/9012 [==============================] - 4s 478us/step - loss: 0.0915 - acc:
0.9760 - val_loss: 21.0335 - val_acc: 0.5010
Epoch 112/150
9012/9012 [==============================] - 4s 491us/step - loss: 0.0590 - acc:
0.9828 - val_loss: 23.3977 - val_acc: 0.4750
Epoch 113/150
9012/9012 [==============================] - 4s 482us/step - loss: 0.0866 - acc:
0.9769 - val_loss: 19.9667 - val_acc: 0.4990
Epoch 114/150
9012/9012 [==============================] - 5s 573us/step - loss: 0.0817 - acc:
0.9781 - val_loss: 16.2853 - val_acc: 0.4042
Epoch 115/150
9012/9012 [==============================] - 5s 572us/step - loss: 0.0793 - acc:
0.9811 - val_loss: 17.0382 - val_acc: 0.4092
Epoch 116/150
9012/9012 [==============================] - 5s 505us/step - loss: 0.0900 - acc:
0.9775 - val_loss: 13.2246 - val_acc: 0.3932
Epoch 117/150
9012/9012 [==============================] - 4s 491us/step - loss: 0.0418 - acc:
0.9879 - val_loss: 38.9175 - val_acc: 0.4621
Epoch 118/150
9012/9012 [==============================] - 4s 483us/step - loss: 0.0994 - acc:
0.9730 - val_loss: 16.4758 - val_acc: 0.4441
Epoch 119/150
9012/9012 [==============================] - 4s 474us/step - loss: 0.0657 - acc:
0.9820 - val_loss: 27.6680 - val_acc: 0.3563
Epoch 120/150
9012/9012 [==============================] - 4s 468us/step - loss: 0.0994 - acc:
0.9748 - val_loss: 17.7414 - val_acc: 0.4880
Epoch 121/150
9012/9012 [==============================] - 4s 484us/step - loss: 0.0671 - acc:
0.9818 - val_loss: 25.1144 - val_acc: 0.5130
Epoch 122/150
9012/9012 [==============================] - 4s 492us/step - loss: 0.0776 - acc:
0.9780 - val_loss: 20.9311 - val_acc: 0.4232
Epoch 123/150
9012/9012 [==============================] - 4s 479us/step - loss: 0.0753 - acc:
0.9819 - val_loss: 22.0477 - val_acc: 0.4870
Epoch 124/150
9012/9012 [==============================] - 4s 480us/step - loss: 0.0573 - acc:
0.9835 - val_loss: 23.1473 - val_acc: 0.4431
Epoch 125/150
9012/9012 [==============================] - 4s 471us/step - loss: 0.0894 - acc:
0.9789 - val_loss: 29.9027 - val_acc: 0.4800
Epoch 126/150
9012/9012 [==============================] - 4s 466us/step - loss: 0.0871 - acc:
0.9784 - val_loss: 11.3723 - val_acc: 0.5838
```

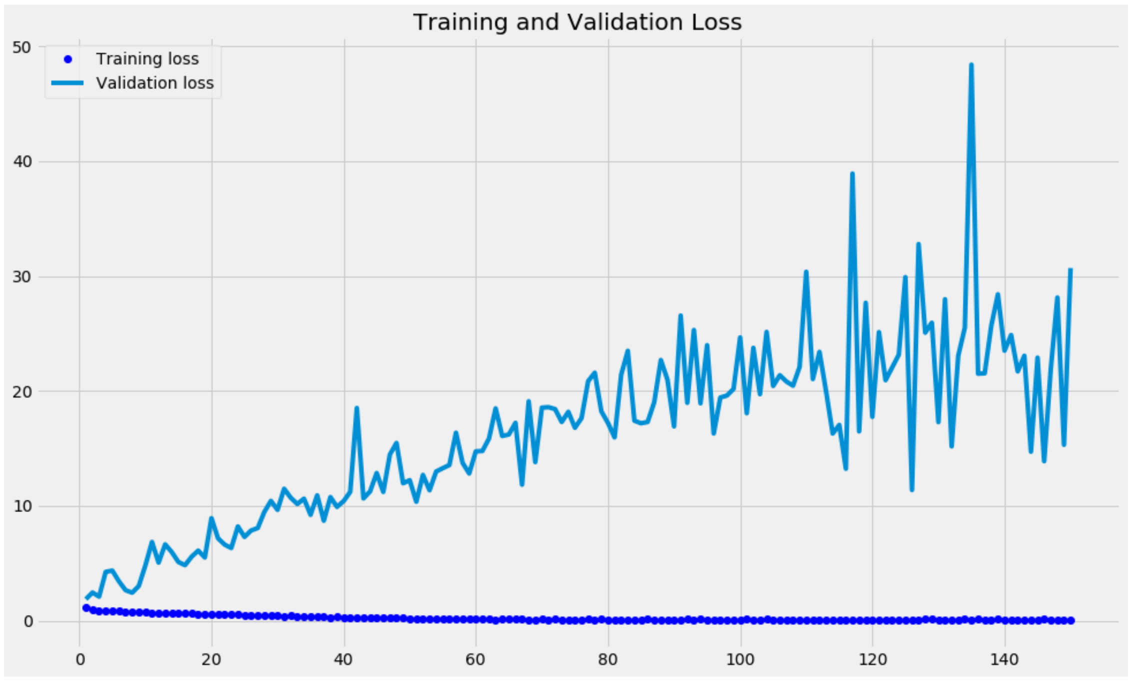
```
Epoch 127/150
9012/9012 [==============================] - 4s 471us/step - loss: 0.0820 - acc:
0.9747 - val_loss: 32.7794 - val_acc: 0.5319
Epoch 128/150
9012/9012 [==============================] - 4s 484us/step - loss: 0.1248 - acc:
0.9703 - val_loss: 25.0672 - val_acc: 0.4491
Epoch 129/150
9012/9012 [==============================] - 4s 475us/step - loss: 0.1397 - acc:
0.9749 - val_loss: 25.9352 - val_acc: 0.5150
Epoch 130/150
9012/9012 [==============================] - 4s 466us/step - loss: 0.0648 - acc:
0.9824 - val_loss: 17.2825 - val_acc: 0.4481
Epoch 131/150
9012/9012 [==============================] - 4s 468us/step - loss: 0.0680 - acc:
0.9818 - val_loss: 27.9777 - val_acc: 0.5000
Epoch 132/150
9012/9012 [==============================] - 4s 479us/step - loss: 0.1051 - acc:
0.9781 - val_loss: 15.1709 - val_acc: 0.4501
Epoch 133/150
9012/9012 [==============================] - 4s 473us/step - loss: 0.0413 - acc:
0.9886 - val_loss: 23.0580 - val_acc: 0.3992
Epoch 134/150
9012/9012 [==============================] - 4s 470us/step - loss: 0.1150 - acc:
0.9688 - val_loss: 25.5083 - val_acc: 0.4671
Epoch 135/150
9012/9012 [==============================] - 4s 486us/step - loss: 0.0377 - acc:
0.9892 - val_loss: 48.3930 - val_acc: 0.4032
Epoch 136/150
9012/9012 [==============================] - 4s 490us/step - loss: 0.1388 - acc:
0.9704 - val_loss: 21.4958 - val_acc: 0.4621
Epoch 137/150
9012/9012 [==============================] - 4s 477us/step - loss: 0.0692 - acc:
0.9814 - val_loss: 21.5264 - val_acc: 0.4451
Epoch 138/150
9012/9012 [==============================] - 4s 476us/step - loss: 0.0722 - acc:
0.9812 - val_loss: 25.6761 - val_acc: 0.4581
Epoch 139/150
9012/9012 [==============================] - 5s 504us/step - loss: 0.1287 - acc:
0.9709 - val_loss: 28.4081 - val_acc: 0.4641
Epoch 140/150
9012/9012 [==============================] - 4s 472us/step - loss: 0.0782 - acc:
0.9811 - val_loss: 23.5003 - val_acc: 0.4741
Epoch 141/150
9012/9012 [==============================] - 4s 472us/step - loss: 0.0765 - acc:
0.9799 - val_loss: 24.8655 - val_acc: 0.3653
Epoch 142/150
9012/9012 [==============================] - 4s 489us/step - loss: 0.0714 - acc:
0.9828 - val_loss: 21.6981 - val_acc: 0.5170
```

```
Epoch 143/150
9012/9012 [==============================] - 5s 502us/step - loss: 0.0872 - acc:
0.9804 - val_loss: 23.0624 - val_acc: 0.4920
Epoch 144/150
9012/9012 [==============================] - 4s 476us/step - loss: 0.0971 - acc:
0.9769 - val_loss: 14.7084 - val_acc: 0.5449
Epoch 145/150
9012/9012 [==============================] - 5s 516us/step - loss: 0.0598 - acc:
0.9845 - val_loss: 22.8964 - val_acc: 0.5369
Epoch 146/150
9012/9012 [==============================] - 4s 495us/step - loss: 0.1249 - acc:
0.9669 - val_loss: 13.8813 - val_acc: 0.5848
Epoch 147/150
9012/9012 [==============================] - 4s 467us/step - loss: 0.0796 - acc:
0.9783 - val_loss: 21.8922 - val_acc: 0.4770
Epoch 148/150
9012/9012 [==============================] - 4s 469us/step - loss: 0.0592 - acc:
0.9857 - val_loss: 28.1261 - val_acc: 0.4172
Epoch 149/150
9012/9012 [==============================] - 4s 471us/step - loss: 0.1063 - acc:
0.9761 - val_loss: 15.3084 - val_acc: 0.4581
Epoch 150/150
9012/9012 [==============================] - 4s 467us/step - loss: 0.0751 - acc:
0.9785 - val_loss: 30.6925 - val_acc: 0.3842
```

[354]:
```
fig = plt.figure(figsize=(15, 20))

accuracy = history_2.history['acc']
val_accuracy = history_2.history['val_acc']
loss = history_2.history['loss']
val_loss = history_2.history['val_loss']

plt.subplot(2,1,1)
epochs = range(1, len(accuracy) + 1)
plt.plot(epochs, accuracy, 'bo', label='Training acc')
plt.plot(epochs, val_accuracy, label='Validation acc')
plt.title('Training and Validation Accuracy')
plt.legend()
fig1 = plt.figure()
fig1.savefig('train_val_acc_s.png')

fig = plt.figure(figsize=(15, 20))
plt.subplot(2,1,2)
plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, label='Validation loss')
plt.title('Training and Validation Loss')
plt.legend()
plt.figure()
plt.show();
fig.savefig('train_val_loss_s.png')
```



```
<Figure size 432x288 with 0 Axes>
```



```
<Figure size 432x288 with 0 Axes>
```

Here, accuracy decreased compared to the baseline model and loss increased significantly. It performed well on the training data but didn't do good enough on validation dataset. But then again, class imbalance was not considered. For this images with higher resolution can be considered.

[355]:
```
# balancing class for images with higher resolution

metadata_image = metadata
metadata_image.head()
```

[355]:
```
     lesion_id      image_id   dx dx_type   age   sex localization  label
0  HAM_0000118  ISIC_0027419  bkl   histo  80.0  male        scalp      2
1  HAM_0000118  ISIC_0025030  bkl   histo  80.0  male        scalp      2
2  HAM_0002730  ISIC_0026769  bkl   histo  80.0  male        scalp      2
3  HAM_0002730  ISIC_0025661  bkl   histo  80.0  male        scalp      2
4  HAM_0001466  ISIC_0031633  bkl   histo  75.0  male          ear      2
```

[356]:
```
# there is a class imbalance, 6,705 instances in melanocytic nevi

metadata_image.label.value_counts()
```

[356]:
```
5    6705
4    1113
2    1099
1     514
0     327
6     142
3     115
Name: label, dtype: int64
```

[357]:
```
# feeding into model

X = metadata_image
y = metadata_image['label']


# resolving the issue of class imabalance using the random under sampler

rus = RandomUnderSampler(random_state=0)
rus.fit(X, y)
X_resampled, y_resampled = rus.fit_resample(X, y)
```

```
# size of our X and y vairables after adjusting

print(X_resampled.shape)
print(y_resampled.shape)

print(X_resampled.iloc[0,:])
```

```
(805, 8)
(805,)
lesion_id        HAM_0002734
image_id        ISIC_0028190
dx                     akiec
dx_type                histo
age                       45
sex                     male
localization            face
label                      0
Name: 0, dtype: object
```

[358]:
```
# checking the number of images in each class, our classes are now balanced

np.unique(y_resampled, return_counts=True)
```

[358]:
```
(array([0, 1, 2, 3, 4, 5, 6]), array([115, 115, 115, 115, 115, 115, 115]))
```

[359]:
```
# Assinging X, and y for model

image_ids = X_resampled.iloc[:,1]
y = y_resampled
```

[360]:
```
# loading 805 images


image_array = []
num_images = 805
i=0



for ID in tqdm(image_ids):

    im = f'/Users/ragnar/Desktop/HAM10000_images_part_1/{ID}.jpg'
    im = np.asarray(imageio.imread(im))
    image_array.append(im)

    i +=1
    if i == num_images:
        break
```

```
 99%|     | 795/805 [00:09<00:00, 84.39it/s]
```

[361]:
```
# reducing total intensity of each pixel to range 0 to 1 to reduce computation␣
 ↪time

x_images = np.asarray(image_array) / 255
print(x_images.shape)
print(y.shape)
```

```
(805, 450, 600, 3)
(805,)
```

[362]:
```
# Building the same baseline architecture we used earlier with the 28x28 images

cnn = Sequential()

cnn.add(MaxPool2D((4, 4), input_shape=(450, 600,3)))

cnn.add(Conv2D(32, (3, 3), strides = (3,3), padding ='same', activation='relu'))
cnn.add(Conv2D(32, (3, 3), activation='relu', padding='same'))
cnn.add(Dropout(0.25))

cnn.add(Conv2D(64, (3, 3), activation='relu',padding ='same'))
cnn.add(Conv2D(64, (3, 3), activation='relu',padding ='same'))
cnn.add(MaxPool2D((2, 2)))
cnn.add(Dropout(0.40))
cnn.add(Flatten())

cnn.add(Dense(128, activation='relu'))
cnn.add(Dense(7, activation='softmax'))

print(cnn.summary())
```

```
Model: "sequential_18"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
max_pooling2d_49 (MaxPooling (None, 112, 150, 3)       0
_________________________________________________________________
conv2d_60 (Conv2D)           (None, 38, 50, 32)        896
_________________________________________________________________
conv2d_61 (Conv2D)           (None, 38, 50, 32)        9248
_________________________________________________________________
dropout_18 (Dropout)         (None, 38, 50, 32)        0
_________________________________________________________________
conv2d_62 (Conv2D)           (None, 38, 50, 64)        18496
_________________________________________________________________
conv2d_63 (Conv2D)           (None, 38, 50, 64)        36928
```

```
_________________________________________________________________
max_pooling2d_50 (MaxPooling (None, 19, 25, 64)        0
_________________________________________________________________
dropout_19 (Dropout)         (None, 19, 25, 64)        0
_________________________________________________________________
flatten_18 (Flatten)         (None, 30400)             0
_________________________________________________________________
dense_39 (Dense)             (None, 128)               3891328
_________________________________________________________________
dense_40 (Dense)             (None, 7)                 903
=================================================================
Total params: 3,957,799
Trainable params: 3,957,799
Non-trainable params: 0
_________________________________________________________________
None
```

[363]:
```python
# Complied Model

cnn.compile('rmsprop', loss='categorical_crossentropy', metrics=['acc'])
history_3 = cnn.fit(
    x_images, to_categorical(y),
    epochs=150, batch_size=200,
    validation_split=0.1,
    verbose=1,
    callbacks=[ModelCheckpoint('skin-cancer.h5', save_best_only=True)]
)
```

```
Train on 724 samples, validate on 81 samples
Epoch 1/150
724/724 [==============================] - 25s 34ms/step - loss: 3.9112 - acc:
0.1685 - val_loss: 1.9671 - val_acc: 0.0000e+00
Epoch 2/150
724/724 [==============================] - 33s 46ms/step - loss: 1.9219 - acc:
0.1561 - val_loss: 2.0599 - val_acc: 0.0000e+00
Epoch 3/150
724/724 [==============================] - 31s 42ms/step - loss: 1.9032 - acc:
0.1464 - val_loss: 2.5246 - val_acc: 0.0000e+00
Epoch 4/150
724/724 [==============================] - 32s 45ms/step - loss: 1.8892 - acc:
0.1961 - val_loss: 2.4834 - val_acc: 0.0000e+00
Epoch 5/150
724/724 [==============================] - 31s 43ms/step - loss: 1.9026 - acc:
0.1865 - val_loss: 2.5329 - val_acc: 0.0000e+00
Epoch 6/150
724/724 [==============================] - 33s 45ms/step - loss: 1.9001 - acc:
0.2459 - val_loss: 3.1524 - val_acc: 0.0000e+00
Epoch 7/150
```

```
724/724 [==============================] - 33s 46ms/step - loss: 1.8861 - acc:
0.1878 - val_loss: 2.5768 - val_acc: 0.0000e+00
Epoch 8/150
724/724 [==============================] - 31s 42ms/step - loss: 1.8797 - acc:
0.1920 - val_loss: 2.5993 - val_acc: 0.0000e+00
Epoch 9/150
724/724 [==============================] - 32s 45ms/step - loss: 1.8594 - acc:
0.2541 - val_loss: 2.5368 - val_acc: 0.0000e+00
Epoch 10/150
724/724 [==============================] - 31s 42ms/step - loss: 1.9268 - acc:
0.2072 - val_loss: 2.9941 - val_acc: 0.0000e+00
Epoch 11/150
724/724 [==============================] - 32s 44ms/step - loss: 1.8696 - acc:
0.2500 - val_loss: 3.2532 - val_acc: 0.0000e+00
Epoch 12/150
724/724 [==============================] - 33s 46ms/step - loss: 1.8540 - acc:
0.2307 - val_loss: 2.9521 - val_acc: 0.0000e+00
Epoch 13/150
724/724 [==============================] - 36s 50ms/step - loss: 1.8189 - acc:
0.2431 - val_loss: 3.0867 - val_acc: 0.0000e+00
Epoch 14/150
724/724 [==============================] - 33s 46ms/step - loss: 1.7726 - acc:
0.2693 - val_loss: 2.5628 - val_acc: 0.0000e+00
Epoch 15/150
724/724 [==============================] - 32s 45ms/step - loss: 1.7302 - acc:
0.3080 - val_loss: 3.2072 - val_acc: 0.0000e+00
Epoch 16/150
724/724 [==============================] - 33s 46ms/step - loss: 1.7727 - acc:
0.2638 - val_loss: 2.6085 - val_acc: 0.0000e+00
Epoch 17/150
724/724 [==============================] - 30s 41ms/step - loss: 1.6991 - acc:
0.2914 - val_loss: 3.5988 - val_acc: 0.0000e+00
Epoch 18/150
724/724 [==============================] - 30s 42ms/step - loss: 1.8727 - acc:
0.2210 - val_loss: 2.7844 - val_acc: 0.0000e+00
Epoch 19/150
724/724 [==============================] - 32s 44ms/step - loss: 1.7613 - acc:
0.2707 - val_loss: 2.7593 - val_acc: 0.0000e+00
Epoch 20/150
724/724 [==============================] - 32s 45ms/step - loss: 1.6906 - acc:
0.3315 - val_loss: 2.6541 - val_acc: 0.0000e+00
Epoch 21/150
724/724 [==============================] - 32s 44ms/step - loss: 1.6853 - acc:
0.2970 - val_loss: 3.0498 - val_acc: 0.0000e+00
Epoch 22/150
724/724 [==============================] - 31s 43ms/step - loss: 1.7112 - acc:
0.2983 - val_loss: 2.4849 - val_acc: 0.0000e+00
Epoch 23/150
```

```
724/724 [==============================] - 31s 43ms/step - loss: 1.7698 - acc:
0.2887 - val_loss: 2.9606 - val_acc: 0.0000e+00
Epoch 24/150
724/724 [==============================] - 32s 44ms/step - loss: 1.6499 - acc:
0.2928 - val_loss: 2.5305 - val_acc: 0.0000e+00
Epoch 25/150
724/724 [==============================] - 33s 45ms/step - loss: 1.5740 - acc:
0.3646 - val_loss: 2.2830 - val_acc: 0.0123
Epoch 26/150
724/724 [==============================] - 33s 46ms/step - loss: 1.6739 - acc:
0.2942 - val_loss: 2.1249 - val_acc: 0.0123
Epoch 27/150
724/724 [==============================] - 34s 47ms/step - loss: 1.5485 - acc:
0.3591 - val_loss: 2.7647 - val_acc: 0.0000e+00
Epoch 28/150
724/724 [==============================] - 31s 43ms/step - loss: 1.5607 - acc:
0.3467 - val_loss: 2.5537 - val_acc: 0.0617
Epoch 29/150
724/724 [==============================] - 35s 48ms/step - loss: 1.5279 - acc:
0.4047 - val_loss: 3.8590 - val_acc: 0.0000e+00
Epoch 30/150
724/724 [==============================] - 35s 48ms/step - loss: 1.6069 - acc:
0.3343 - val_loss: 2.8887 - val_acc: 0.0617
Epoch 31/150
724/724 [==============================] - 34s 46ms/step - loss: 1.5437 - acc:
0.3633 - val_loss: 3.2986 - val_acc: 0.0370
Epoch 32/150
724/724 [==============================] - 35s 48ms/step - loss: 1.4695 - acc:
0.4088 - val_loss: 3.1629 - val_acc: 0.0741
Epoch 33/150
724/724 [==============================] - 30s 42ms/step - loss: 1.4758 - acc:
0.3992 - val_loss: 2.1105 - val_acc: 0.3457
Epoch 34/150
724/724 [==============================] - 34s 47ms/step - loss: 1.4640 - acc:
0.4047 - val_loss: 2.3797 - val_acc: 0.1605
Epoch 35/150
724/724 [==============================] - 31s 43ms/step - loss: 1.4902 - acc:
0.4116 - val_loss: 1.8734 - val_acc: 0.2716
Epoch 36/150
724/724 [==============================] - 32s 44ms/step - loss: 1.5681 - acc:
0.3494 - val_loss: 1.8227 - val_acc: 0.5309
Epoch 37/150
724/724 [==============================] - 34s 47ms/step - loss: 1.3932 - acc:
0.4227 - val_loss: 3.6090 - val_acc: 0.0247
Epoch 38/150
724/724 [==============================] - 34s 47ms/step - loss: 1.4206 - acc:
0.4530 - val_loss: 2.2070 - val_acc: 0.3827
Epoch 39/150
```

```
724/724 [==============================] - 34s 47ms/step - loss: 1.3322 - acc:
0.4710 - val_loss: 1.6792 - val_acc: 0.5556
Epoch 40/150
724/724 [==============================] - 37s 51ms/step - loss: 1.3030 - acc:
0.4986 - val_loss: 1.4538 - val_acc: 0.6296
Epoch 41/150
724/724 [==============================] - 33s 45ms/step - loss: 1.3218 - acc:
0.4751 - val_loss: 1.8289 - val_acc: 0.5185
Epoch 42/150
724/724 [==============================] - 35s 49ms/step - loss: 1.4026 - acc:
0.4682 - val_loss: 1.3237 - val_acc: 0.6667
Epoch 43/150
724/724 [==============================] - 36s 50ms/step - loss: 1.2626 - acc:
0.5055 - val_loss: 2.4016 - val_acc: 0.3210
Epoch 44/150
724/724 [==============================] - 37s 50ms/step - loss: 1.4445 - acc:
0.4240 - val_loss: 2.6184 - val_acc: 0.3210
Epoch 45/150
724/724 [==============================] - 35s 49ms/step - loss: 1.1518 - acc:
0.5525 - val_loss: 2.4231 - val_acc: 0.3210
Epoch 46/150
724/724 [==============================] - 34s 46ms/step - loss: 1.1706 - acc:
0.5497 - val_loss: 4.1400 - val_acc: 0.0247
Epoch 47/150
724/724 [==============================] - 34s 48ms/step - loss: 1.2975 - acc:
0.5262 - val_loss: 2.0548 - val_acc: 0.4198
Epoch 48/150
724/724 [==============================] - 37s 51ms/step - loss: 1.2411 - acc:
0.5276 - val_loss: 2.3352 - val_acc: 0.4691
Epoch 49/150
724/724 [==============================] - 31s 43ms/step - loss: 1.1820 - acc:
0.5456 - val_loss: 1.9761 - val_acc: 0.4691
Epoch 50/150
724/724 [==============================] - 32s 45ms/step - loss: 1.1478 - acc:
0.5608 - val_loss: 1.7556 - val_acc: 0.5926
Epoch 51/150
724/724 [==============================] - 34s 47ms/step - loss: 0.9796 - acc:
0.6202 - val_loss: 2.5413 - val_acc: 0.4444
Epoch 52/150
724/724 [==============================] - 31s 43ms/step - loss: 1.2222 - acc:
0.5331 - val_loss: 1.9799 - val_acc: 0.5309
Epoch 53/150
724/724 [==============================] - 33s 45ms/step - loss: 0.9288 - acc:
0.6685 - val_loss: 3.3929 - val_acc: 0.2099
Epoch 54/150
724/724 [==============================] - 34s 47ms/step - loss: 1.0796 - acc:
0.5718 - val_loss: 4.0103 - val_acc: 0.1481
Epoch 55/150
```

```
724/724 [==============================] - 34s 47ms/step - loss: 1.0266 - acc:
0.6064 - val_loss: 2.1048 - val_acc: 0.5432
Epoch 56/150
724/724 [==============================] - 30s 42ms/step - loss: 1.0119 - acc:
0.6229 - val_loss: 2.9517 - val_acc: 0.4198
Epoch 57/150
724/724 [==============================] - 35s 49ms/step - loss: 0.9446 - acc:
0.6588 - val_loss: 2.5755 - val_acc: 0.3457
Epoch 58/150
724/724 [==============================] - 35s 48ms/step - loss: 0.9037 - acc:
0.6616 - val_loss: 1.2835 - val_acc: 0.7160
Epoch 59/150
724/724 [==============================] - 34s 48ms/step - loss: 0.8638 - acc:
0.6657 - val_loss: 2.9426 - val_acc: 0.2840
Epoch 60/150
724/724 [==============================] - 35s 48ms/step - loss: 1.0666 - acc:
0.5843 - val_loss: 3.3094 - val_acc: 0.1728
Epoch 61/150
724/724 [==============================] - 32s 44ms/step - loss: 0.9750 - acc:
0.6381 - val_loss: 2.5735 - val_acc: 0.3827
Epoch 62/150
724/724 [==============================] - 34s 47ms/step - loss: 0.7554 - acc:
0.7113 - val_loss: 2.0333 - val_acc: 0.5679
Epoch 63/150
724/724 [==============================] - 33s 46ms/step - loss: 0.7462 - acc:
0.7030 - val_loss: 4.6743 - val_acc: 0.2840
Epoch 64/150
724/724 [==============================] - 32s 44ms/step - loss: 1.3019 - acc:
0.5608 - val_loss: 3.8217 - val_acc: 0.1975
Epoch 65/150
724/724 [==============================] - 31s 43ms/step - loss: 0.7522 - acc:
0.7320 - val_loss: 2.6870 - val_acc: 0.4938
Epoch 66/150
724/724 [==============================] - 34s 47ms/step - loss: 0.6359 - acc:
0.7887 - val_loss: 4.0107 - val_acc: 0.3210
Epoch 67/150
724/724 [==============================] - 36s 49ms/step - loss: 0.7043 - acc:
0.7334 - val_loss: 3.3422 - val_acc: 0.2593
Epoch 68/150
724/724 [==============================] - 35s 48ms/step - loss: 0.7469 - acc:
0.7238 - val_loss: 4.2367 - val_acc: 0.2963
Epoch 69/150
724/724 [==============================] - 30s 41ms/step - loss: 0.6184 - acc:
0.7804 - val_loss: 3.4650 - val_acc: 0.2346
Epoch 70/150
724/724 [==============================] - 33s 46ms/step - loss: 0.9081 - acc:
0.6878 - val_loss: 2.5299 - val_acc: 0.4815
Epoch 71/150
```

```
724/724 [==============================] - 33s 45ms/step - loss: 0.6239 - acc:
0.7597 - val_loss: 2.8757 - val_acc: 0.5185
Epoch 72/150
724/724 [==============================] - 33s 45ms/step - loss: 0.5881 - acc:
0.7928 - val_loss: 2.1990 - val_acc: 0.6296
Epoch 73/150
724/724 [==============================] - 34s 47ms/step - loss: 0.6386 - acc:
0.7680 - val_loss: 2.5708 - val_acc: 0.5679
Epoch 74/150
724/724 [==============================] - 31s 43ms/step - loss: 0.5720 - acc:
0.7749 - val_loss: 3.8963 - val_acc: 0.3457
Epoch 75/150
724/724 [==============================] - 32s 44ms/step - loss: 1.2874 - acc:
0.6146 - val_loss: 1.9611 - val_acc: 0.5679
Epoch 76/150
724/724 [==============================] - 38s 52ms/step - loss: 0.5540 - acc:
0.8260 - val_loss: 4.0162 - val_acc: 0.3457
Epoch 77/150
724/724 [==============================] - 32s 44ms/step - loss: 0.4491 - acc:
0.8439 - val_loss: 2.8249 - val_acc: 0.6173
Epoch 78/150
724/724 [==============================] - 33s 46ms/step - loss: 0.4826 - acc:
0.8246 - val_loss: 3.0660 - val_acc: 0.5926
Epoch 79/150
724/724 [==============================] - 33s 46ms/step - loss: 0.5232 - acc:
0.8066 - val_loss: 1.8679 - val_acc: 0.6296
Epoch 80/150
724/724 [==============================] - 34s 46ms/step - loss: 0.3707 - acc:
0.8771 - val_loss: 4.0817 - val_acc: 0.3951
Epoch 81/150
724/724 [==============================] - 31s 43ms/step - loss: 0.4107 - acc:
0.8384 - val_loss: 5.6293 - val_acc: 0.1605
Epoch 82/150
724/724 [==============================] - 33s 46ms/step - loss: 0.5075 - acc:
0.8039 - val_loss: 7.0948 - val_acc: 0.0741
Epoch 83/150
724/724 [==============================] - 36s 49ms/step - loss: 0.4260 - acc:
0.8370 - val_loss: 2.5214 - val_acc: 0.5679
Epoch 84/150
724/724 [==============================] - 35s 48ms/step - loss: 0.5007 - acc:
0.8425 - val_loss: 3.4278 - val_acc: 0.5185
Epoch 85/150
724/724 [==============================] - 35s 48ms/step - loss: 0.2768 - acc:
0.9282 - val_loss: 7.2120 - val_acc: 0.3210
Epoch 86/150
724/724 [==============================] - 31s 43ms/step - loss: 0.6651 - acc:
0.7638 - val_loss: 3.1133 - val_acc: 0.5309
Epoch 87/150
```

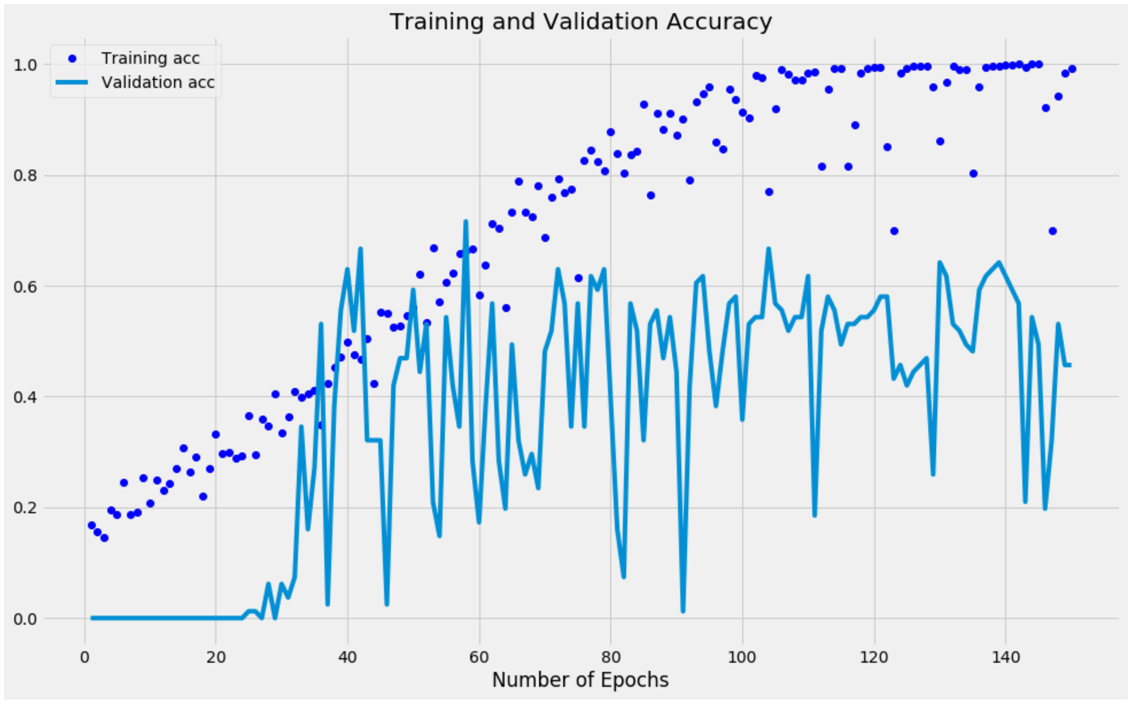
```
724/724 [==============================] - 33s 46ms/step - loss: 0.2833 - acc:
0.9102 - val_loss: 4.5726 - val_acc: 0.5556
Epoch 88/150
724/724 [==============================] - 33s 46ms/step - loss: 0.3341 - acc:
0.8826 - val_loss: 4.6691 - val_acc: 0.4691
Epoch 89/150
724/724 [==============================] - 31s 43ms/step - loss: 0.2544 - acc:
0.9116 - val_loss: 4.0601 - val_acc: 0.5432
Epoch 90/150
724/724 [==============================] - 36s 50ms/step - loss: 0.4021 - acc:
0.8715 - val_loss: 4.4547 - val_acc: 0.4444
Epoch 91/150
724/724 [==============================] - 35s 49ms/step - loss: 0.3766 - acc:
0.9006 - val_loss: 8.5494 - val_acc: 0.0123
Epoch 92/150
724/724 [==============================] - 34s 48ms/step - loss: 0.7312 - acc:
0.7901 - val_loss: 3.8574 - val_acc: 0.4198
Epoch 93/150
724/724 [==============================] - 31s 43ms/step - loss: 0.2262 - acc:
0.9323 - val_loss: 2.7855 - val_acc: 0.6049
Epoch 94/150
724/724 [==============================] - 33s 46ms/step - loss: 0.1570 - acc:
0.9475 - val_loss: 2.8919 - val_acc: 0.6173
Epoch 95/150
724/724 [==============================] - 34s 47ms/step - loss: 0.1286 - acc:
0.9599 - val_loss: 4.5730 - val_acc: 0.4815
Epoch 96/150
724/724 [==============================] - 34s 47ms/step - loss: 0.4398 - acc:
0.8591 - val_loss: 4.2089 - val_acc: 0.3827
Epoch 97/150
724/724 [==============================] - 34s 47ms/step - loss: 0.4325 - acc:
0.8467 - val_loss: 4.1053 - val_acc: 0.4815
Epoch 98/150
724/724 [==============================] - 33s 45ms/step - loss: 0.1525 - acc:
0.9558 - val_loss: 3.8164 - val_acc: 0.5679
Epoch 99/150
724/724 [==============================] - 34s 47ms/step - loss: 0.1688 - acc:
0.9365 - val_loss: 3.6805 - val_acc: 0.5802
Epoch 100/150
724/724 [==============================] - 33s 45ms/step - loss: 0.2801 - acc:
0.9130 - val_loss: 4.7730 - val_acc: 0.3580
Epoch 101/150
724/724 [==============================] - 32s 45ms/step - loss: 0.3673 - acc:
0.9033 - val_loss: 3.9120 - val_acc: 0.5309
Epoch 102/150
724/724 [==============================] - 34s 46ms/step - loss: 0.0804 - acc:
0.9807 - val_loss: 4.2264 - val_acc: 0.5432
Epoch 103/150
```

```
724/724 [==============================] - 33s 46ms/step - loss: 0.0773 - acc:
0.9751 - val_loss: 3.2866 - val_acc: 0.5432
Epoch 104/150
724/724 [==============================] - 33s 46ms/step - loss: 0.6563 - acc:
0.7693 - val_loss: 1.9434 - val_acc: 0.6667
Epoch 105/150
724/724 [==============================] - 32s 44ms/step - loss: 0.2816 - acc:
0.9185 - val_loss: 3.8326 - val_acc: 0.5679
Epoch 106/150
724/724 [==============================] - 33s 46ms/step - loss: 0.0576 - acc:
0.9903 - val_loss: 4.2903 - val_acc: 0.5556
Epoch 107/150
724/724 [==============================] - 34s 46ms/step - loss: 0.0758 - acc:
0.9820 - val_loss: 5.0179 - val_acc: 0.5185
Epoch 108/150
724/724 [==============================] - 33s 46ms/step - loss: 0.1050 - acc:
0.9710 - val_loss: 4.3224 - val_acc: 0.5432
Epoch 109/150
724/724 [==============================] - 36s 50ms/step - loss: 0.0939 - acc:
0.9724 - val_loss: 4.6504 - val_acc: 0.5432
Epoch 110/150
724/724 [==============================] - 33s 46ms/step - loss: 0.0539 - acc:
0.9834 - val_loss: 4.8427 - val_acc: 0.6173
Epoch 111/150
724/724 [==============================] - 37s 51ms/step - loss: 0.0478 - acc:
0.9862 - val_loss: 9.1936 - val_acc: 0.1852
Epoch 112/150
724/724 [==============================] - 39s 54ms/step - loss: 0.5638 - acc:
0.8149 - val_loss: 5.2563 - val_acc: 0.5185
Epoch 113/150
724/724 [==============================] - 35s 48ms/step - loss: 0.1514 - acc:
0.9558 - val_loss: 4.0491 - val_acc: 0.5802
Epoch 114/150
724/724 [==============================] - 36s 50ms/step - loss: 0.0461 - acc:
0.9931 - val_loss: 4.9099 - val_acc: 0.5556
Epoch 115/150
724/724 [==============================] - 38s 52ms/step - loss: 0.0259 - acc:
0.9917 - val_loss: 6.6606 - val_acc: 0.4938
Epoch 116/150
724/724 [==============================] - 34s 47ms/step - loss: 0.7033 - acc:
0.8163 - val_loss: 4.1453 - val_acc: 0.5309
Epoch 117/150
724/724 [==============================] - 33s 45ms/step - loss: 0.3262 - acc:
0.8895 - val_loss: 5.5625 - val_acc: 0.5309
Epoch 118/150
724/724 [==============================] - 32s 44ms/step - loss: 0.0608 - acc:
0.9848 - val_loss: 5.4080 - val_acc: 0.5432
Epoch 119/150
```

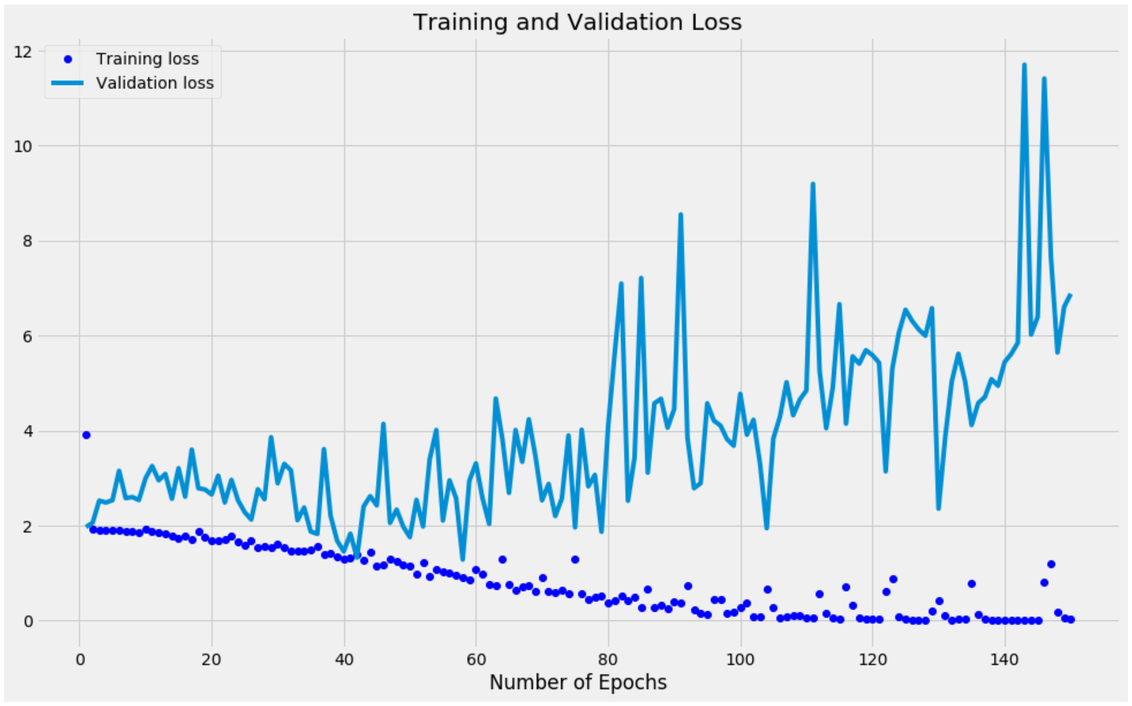
```
724/724 [==============================] - 34s 47ms/step - loss: 0.0302 - acc:
0.9931 - val_loss: 5.6919 - val_acc: 0.5432
Epoch 120/150
724/724 [==============================] - 31s 42ms/step - loss: 0.0225 - acc:
0.9945 - val_loss: 5.5882 - val_acc: 0.5556
Epoch 121/150
724/724 [==============================] - 32s 44ms/step - loss: 0.0228 - acc:
0.9945 - val_loss: 5.4262 - val_acc: 0.5802
Epoch 122/150
724/724 [==============================] - 33s 46ms/step - loss: 0.6125 - acc:
0.8508 - val_loss: 3.1402 - val_acc: 0.5802
Epoch 123/150
724/724 [==============================] - 32s 44ms/step - loss: 0.8750 - acc:
0.6989 - val_loss: 5.3001 - val_acc: 0.4321
Epoch 124/150
724/724 [==============================] - 34s 47ms/step - loss: 0.0874 - acc:
0.9834 - val_loss: 6.0619 - val_acc: 0.4568
Epoch 125/150
724/724 [==============================] - 35s 48ms/step - loss: 0.0350 - acc:
0.9917 - val_loss: 6.5436 - val_acc: 0.4198
Epoch 126/150
724/724 [==============================] - 33s 46ms/step - loss: 0.0184 - acc:
0.9972 - val_loss: 6.3084 - val_acc: 0.4444
Epoch 127/150
724/724 [==============================] - 34s 48ms/step - loss: 0.0146 - acc:
0.9972 - val_loss: 6.1281 - val_acc: 0.4568
Epoch 128/150
724/724 [==============================] - 32s 44ms/step - loss: 0.0119 - acc:
0.9959 - val_loss: 5.9966 - val_acc: 0.4691
Epoch 129/150
724/724 [==============================] - 32s 44ms/step - loss: 0.1963 - acc:
0.9599 - val_loss: 6.5765 - val_acc: 0.2593
Epoch 130/150
724/724 [==============================] - 33s 45ms/step - loss: 0.4205 - acc:
0.8605 - val_loss: 2.3550 - val_acc: 0.6420
Epoch 131/150
724/724 [==============================] - 34s 47ms/step - loss: 0.1129 - acc:
0.9669 - val_loss: 3.8508 - val_acc: 0.6173
Epoch 132/150
724/724 [==============================] - 35s 49ms/step - loss: 0.0162 - acc:
0.9972 - val_loss: 5.0358 - val_acc: 0.5309
Epoch 133/150
724/724 [==============================] - 34s 47ms/step - loss: 0.0289 - acc:
0.9903 - val_loss: 5.6177 - val_acc: 0.5185
Epoch 134/150
724/724 [==============================] - 35s 48ms/step - loss: 0.0241 - acc:
0.9903 - val_loss: 5.0463 - val_acc: 0.4938
Epoch 135/150
```

```
724/724 [==============================] - 33s 45ms/step - loss: 0.7843 - acc:
0.8039 - val_loss: 4.1154 - val_acc: 0.4815
Epoch 136/150
724/724 [==============================] - 35s 48ms/step - loss: 0.1223 - acc:
0.9586 - val_loss: 4.5784 - val_acc: 0.5926
Epoch 137/150
724/724 [==============================] - 33s 46ms/step - loss: 0.0225 - acc:
0.9945 - val_loss: 4.7053 - val_acc: 0.6173
Epoch 138/150
724/724 [==============================] - 30s 42ms/step - loss: 0.0125 - acc:
0.9972 - val_loss: 5.0811 - val_acc: 0.6296
Epoch 139/150
724/724 [==============================] - 34s 47ms/step - loss: 0.0137 - acc:
0.9959 - val_loss: 4.9429 - val_acc: 0.6420
Epoch 140/150
724/724 [==============================] - 33s 45ms/step - loss: 0.0062 - acc:
0.9986 - val_loss: 5.4420 - val_acc: 0.6173
Epoch 141/150
724/724 [==============================] - 34s 47ms/step - loss: 0.0079 - acc:
0.9986 - val_loss: 5.6151 - val_acc: 0.5926
Epoch 142/150
724/724 [==============================] - 33s 46ms/step - loss: 0.0066 - acc:
1.0000 - val_loss: 5.8498 - val_acc: 0.5679
Epoch 143/150
724/724 [==============================] - 36s 50ms/step - loss: 0.0155 - acc:
0.9945 - val_loss: 11.7032 - val_acc: 0.2099
Epoch 144/150
724/724 [==============================] - 31s 42ms/step - loss: 0.0059 - acc:
1.0000 - val_loss: 6.0203 - val_acc: 0.5432
Epoch 145/150
724/724 [==============================] - 35s 49ms/step - loss: 0.0027 - acc:
1.0000 - val_loss: 6.3933 - val_acc: 0.4938
Epoch 146/150
724/724 [==============================] - 32s 45ms/step - loss: 0.8120 - acc:
0.9213 - val_loss: 11.4118 - val_acc: 0.1975
Epoch 147/150
724/724 [==============================] - 34s 48ms/step - loss: 1.1915 - acc:
0.7003 - val_loss: 7.6608 - val_acc: 0.3210
Epoch 148/150
724/724 [==============================] - 33s 46ms/step - loss: 0.1762 - acc:
0.9420 - val_loss: 5.6434 - val_acc: 0.5309
Epoch 149/150
724/724 [==============================] - 33s 46ms/step - loss: 0.0670 - acc:
0.9848 - val_loss: 6.6024 - val_acc: 0.4568
Epoch 150/150
724/724 [==============================] - 31s 43ms/step - loss: 0.0392 - acc:
0.9931 - val_loss: 6.8794 - val_acc: 0.4568
```

[364]:
```
fig = plt.figure(figsize=(15, 20))

accuracy = history_3.history['acc']
val_accuracy = history_3.history['val_acc']
loss = history_3.history['loss']
val_loss = history_3.history['val_loss']

plt.subplot(2,1,1)
epochs = range(1, len(accuracy) + 1)
plt.plot(epochs, accuracy, 'bo', label='Training acc')
plt.plot(epochs, val_accuracy, label='Validation acc')
plt.title('Training and Validation Accuracy')
plt.xlabel('Number of Epochs')
plt.legend()
fig1 = plt.figure()
fig1.savefig('train_val_acc_s.png')

fig = plt.figure(figsize=(15, 20))
plt.subplot(2,1,2)
plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, label='Validation loss')
plt.title('Training and Validation Loss')
plt.xlabel('Number of Epochs')
plt.legend()
plt.figure()
plt.show();
fig.savefig('train_val_loss_s.png')
```



```
<Figure size 432x288 with 0 Axes>
```



```
<Figure size 432x288 with 0 Axes>
```

This is more unstable than low resolutaion images and the loss function increased instead of decreasing. Model is overfit on training data.

[426]:
```
# standard scaling the pixels to reduce computation time

x_train_standard, x_test_standard, y_train_standard, y_test_standard =␣
↪train_test_split(x_images, y, test_size=.3)
print(x_train_standard.shape)
print(x_test_standard.shape)
print(y_train_standard.shape)
print(y_test_standard.shape)
```

```
(563, 450, 600, 3)
(242, 450, 600, 3)
(563,)
(242,)
```

[427]:
```
x_train_mean = np.mean(x_train_standard)
x_train_std = np.std(x_train_standard)
x_test_mean = np.mean(x_test_standard)
x_test_std = np.std(x_test_standard)

x_train_standard = (x_train_standard - x_train_mean) / x_train_std
x_test_standard = (x_test_standard - x_test_mean) / x_test_std
```

[428]:
```
x_train_standard.shape
```

[428]:
```
(563, 450, 600, 3)
```

[429]:
```
x_test_standard.shape
```

[429]:
```
(242, 450, 600, 3)
```

[430]:
```
y_test_standard = to_categorical(y_test_standard, num_classes=7)
y_train_standard = to_categorical(y_train_standard, num_classes=7)
```

[431]:
```
y_train_standard.shape
```

[431]:
```
(563, 7)
```

[432]:
```
y_test_standard.shape
```

[432]:
```
(242, 7)
```

[412]:
```
# image augmentation to reduce overfitting

datagen = ImageDataGenerator(
        featurewise_center=False,
        samplewise_center=False,
        featurewise_std_normalization=False,
        samplewise_std_normalization=False,
        zca_whitening=False,
        rotation_range=10,
        zoom_range = 0.1,
        width_shift_range=0.1,
        height_shift_range=0.1,
        horizontal_flip=False,
        vertical_flip=False)

datagen.fit(x_train_standard)
```

[378]:
```
# building an architecture with dropout layers to further reduce overfitting␣
 ↪and increase stability of model
```

```
model_4 = Sequential()

model_4.add(MaxPool2D((4, 4), input_shape=(450, 600,3)))

model_4.add(Conv2D(32, (3, 3), padding ='same', activation='relu'))
model_4.add(Conv2D(32, (3, 3),  padding='same', activation='relu'))
model_4.add(MaxPool2D((2, 2)))
model_4.add(Dropout(0.25))

model_4.add(Conv2D(64, (3, 3), activation='relu',padding ='same'))
model_4.add(Conv2D(64, (3, 3), activation='relu',padding ='same'))
model_4.add(MaxPool2D((2, 2)))
model_4.add(Dropout(0.40))
model_4.add(Flatten())

model_4.add(Dense(128, activation='relu'))
model_4.add(Dropout(0.5))
model_4.add(Dense(7, activation='softmax'))

print(model_4.summary())
```

```
Model: "sequential_19"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
max_pooling2d_51 (MaxPooling (None, 112, 150, 3)       0
_________________________________________________________________
conv2d_64 (Conv2D)           (None, 112, 150, 32)      896
_________________________________________________________________
conv2d_65 (Conv2D)           (None, 112, 150, 32)      9248
_________________________________________________________________
max_pooling2d_52 (MaxPooling (None, 56, 75, 32)        0
_________________________________________________________________
dropout_20 (Dropout)         (None, 56, 75, 32)        0
_________________________________________________________________
conv2d_66 (Conv2D)           (None, 56, 75, 64)        18496
_________________________________________________________________
conv2d_67 (Conv2D)           (None, 56, 75, 64)        36928
_________________________________________________________________
max_pooling2d_53 (MaxPooling (None, 28, 37, 64)        0
_________________________________________________________________
dropout_21 (Dropout)         (None, 28, 37, 64)        0
_________________________________________________________________
flatten_19 (Flatten)         (None, 66304)             0
_________________________________________________________________
dense_41 (Dense)             (None, 128)               8487040
_________________________________________________________________
```

```
dropout_22 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_42 (Dense)             (None, 7)                 903
=================================================================
Total params: 8,553,511
Trainable params: 8,553,511
Non-trainable params: 0
_________________________________________________________________
None
```

[423]:
```
optimizer = keras.optimizers.Adam(lr=0.001, beta_1=.9, beta_2=0.999, epsilon=1,␣
 ↪decay=0.0, amsgrad=False)
```

[424]:
```
# Complied Model

model_4.compile(optimizer=optimizer, loss='categorical_crossentropy',␣
 ↪metrics=['accuracy'])
```

[433]:
```
history_4 = model_4.fit_generator(
    datagen.flow(x_train_standard, y_train_standard,batch_size=10),
    epochs=50,
    validation_data=(x_test_standard, y_test_standard),
    verbose=1,
    steps_per_epoch=x_train_standard.shape[0],
    callbacks=[ModelCheckpoint('skin-cancer-4.h5', save_best_only=True)]
)
```

```
Epoch 1/50
563/563 [==============================] - 379s 673ms/step - loss: 1.8984 -
accuracy: 0.2045 - val_loss: 1.7801 - val_accuracy: 0.3182
Epoch 2/50
563/563 [==============================] - 336s 596ms/step - loss: 1.7445 -
accuracy: 0.3040 - val_loss: 1.6420 - val_accuracy: 0.3843
Epoch 3/50
563/563 [==============================] - 336s 597ms/step - loss: 1.5850 -
accuracy: 0.3806 - val_loss: 1.5279 - val_accuracy: 0.4504
Epoch 4/50
563/563 [==============================] - 340s 604ms/step - loss: 1.4456 -
accuracy: 0.4381 - val_loss: 1.4840 - val_accuracy: 0.4545
Epoch 5/50
563/563 [==============================] - 335s 596ms/step - loss: 1.3993 -
accuracy: 0.4642 - val_loss: 1.4314 - val_accuracy: 0.4380
Epoch 6/50
563/563 [==============================] - 336s 597ms/step - loss: 1.3185 -
accuracy: 0.4868 - val_loss: 1.3869 - val_accuracy: 0.4835
Epoch 7/50
563/563 [==============================] - 339s 602ms/step - loss: 1.2606 -
accuracy: 0.5182 - val_loss: 1.3921 - val_accuracy: 0.4380
```

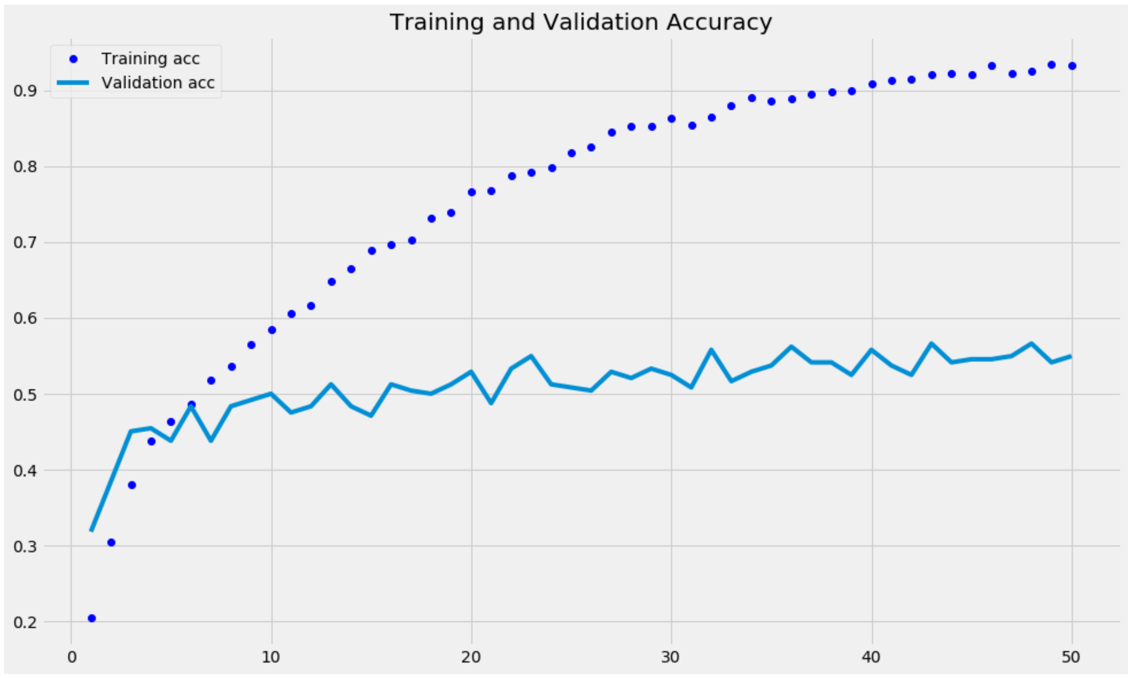
```
Epoch 8/50
563/563 [==============================] - 339s 602ms/step - loss: 1.2187 -
accuracy: 0.5367 - val_loss: 1.3648 - val_accuracy: 0.4835
Epoch 9/50
563/563 [==============================] - 336s 597ms/step - loss: 1.1434 -
accuracy: 0.5656 - val_loss: 1.3177 - val_accuracy: 0.4917
Epoch 10/50
563/563 [==============================] - 337s 598ms/step - loss: 1.0784 -
accuracy: 0.5842 - val_loss: 1.3431 - val_accuracy: 0.5000
Epoch 11/50
563/563 [==============================] - 336s 598ms/step - loss: 1.0471 -
accuracy: 0.6065 - val_loss: 1.3494 - val_accuracy: 0.4752
Epoch 12/50
563/563 [==============================] - 340s 603ms/step - loss: 0.9930 -
accuracy: 0.6163 - val_loss: 1.3885 - val_accuracy: 0.4835
Epoch 13/50
563/563 [==============================] - 336s 597ms/step - loss: 0.9272 -
accuracy: 0.6480 - val_loss: 1.3383 - val_accuracy: 0.5124
Epoch 14/50
563/563 [==============================] - 337s 598ms/step - loss: 0.8860 -
accuracy: 0.6649 - val_loss: 1.3649 - val_accuracy: 0.4835
Epoch 15/50
563/563 [==============================] - 338s 600ms/step - loss: 0.8337 -
accuracy: 0.6887 - val_loss: 1.4335 - val_accuracy: 0.4711
Epoch 16/50
563/563 [==============================] - 338s 600ms/step - loss: 0.8024 -
accuracy: 0.6960 - val_loss: 1.3891 - val_accuracy: 0.5124
Epoch 17/50
563/563 [==============================] - 336s 598ms/step - loss: 0.7735 -
accuracy: 0.7025 - val_loss: 1.3599 - val_accuracy: 0.5041
Epoch 18/50
563/563 [==============================] - 338s 600ms/step - loss: 0.7211 -
accuracy: 0.7307 - val_loss: 1.4306 - val_accuracy: 0.5000
Epoch 19/50
563/563 [==============================] - 344s 611ms/step - loss: 0.6893 -
accuracy: 0.7388 - val_loss: 1.4095 - val_accuracy: 0.5124
Epoch 20/50
563/563 [==============================] - 341s 606ms/step - loss: 0.6369 -
accuracy: 0.7668 - val_loss: 1.4770 - val_accuracy: 0.5289
Epoch 21/50
563/563 [==============================] - 342s 607ms/step - loss: 0.6168 -
accuracy: 0.7674 - val_loss: 1.4847 - val_accuracy: 0.4876
Epoch 22/50
563/563 [==============================] - 342s 607ms/step - loss: 0.5714 -
accuracy: 0.7874 - val_loss: 1.5696 - val_accuracy: 0.5331
Epoch 23/50
563/563 [==============================] - 341s 606ms/step - loss: 0.5529 -
accuracy: 0.7924 - val_loss: 1.6213 - val_accuracy: 0.5496
```

```
Epoch 24/50
563/563 [==============================] - 341s 606ms/step - loss: 0.5304 -
accuracy: 0.7980 - val_loss: 1.6155 - val_accuracy: 0.5124
Epoch 25/50
563/563 [==============================] - 341s 605ms/step - loss: 0.4960 -
accuracy: 0.8178 - val_loss: 1.6645 - val_accuracy: 0.5083
Epoch 26/50
563/563 [==============================] - 341s 606ms/step - loss: 0.4805 -
accuracy: 0.8250 - val_loss: 1.6434 - val_accuracy: 0.5041
Epoch 27/50
563/563 [==============================] - 341s 605ms/step - loss: 0.4353 -
accuracy: 0.8446 - val_loss: 1.6312 - val_accuracy: 0.5289
Epoch 28/50
563/563 [==============================] - 341s 606ms/step - loss: 0.4047 -
accuracy: 0.8520 - val_loss: 1.8791 - val_accuracy: 0.5207
Epoch 29/50
563/563 [==============================] - 341s 606ms/step - loss: 0.4163 -
accuracy: 0.8523 - val_loss: 1.7629 - val_accuracy: 0.5331
Epoch 30/50
563/563 [==============================] - 342s 607ms/step - loss: 0.3810 -
accuracy: 0.8628 - val_loss: 1.9097 - val_accuracy: 0.5248
Epoch 31/50
563/563 [==============================] - 344s 610ms/step - loss: 0.3796 -
accuracy: 0.8536 - val_loss: 1.8484 - val_accuracy: 0.5083
Epoch 32/50
563/563 [==============================] - 341s 605ms/step - loss: 0.3631 -
accuracy: 0.8644 - val_loss: 1.8614 - val_accuracy: 0.5579
Epoch 33/50
563/563 [==============================] - 341s 606ms/step - loss: 0.3332 -
accuracy: 0.8797 - val_loss: 1.8567 - val_accuracy: 0.5165
Epoch 34/50
563/563 [==============================] - 341s 606ms/step - loss: 0.2973 -
accuracy: 0.8899 - val_loss: 2.0237 - val_accuracy: 0.5289
Epoch 35/50
563/563 [==============================] - 341s 606ms/step - loss: 0.3108 -
accuracy: 0.8856 - val_loss: 2.0249 - val_accuracy: 0.5372
Epoch 36/50
563/563 [==============================] - 342s 607ms/step - loss: 0.3100 -
accuracy: 0.8889 - val_loss: 1.9444 - val_accuracy: 0.5620
Epoch 37/50
563/563 [==============================] - 341s 605ms/step - loss: 0.2732 -
accuracy: 0.8949 - val_loss: 2.0604 - val_accuracy: 0.5413
Epoch 38/50
563/563 [==============================] - 341s 606ms/step - loss: 0.2752 -
accuracy: 0.8982 - val_loss: 1.9732 - val_accuracy: 0.5413
Epoch 39/50
563/563 [==============================] - 341s 605ms/step - loss: 0.2805 -
accuracy: 0.8995 - val_loss: 2.0611 - val_accuracy: 0.5248
```

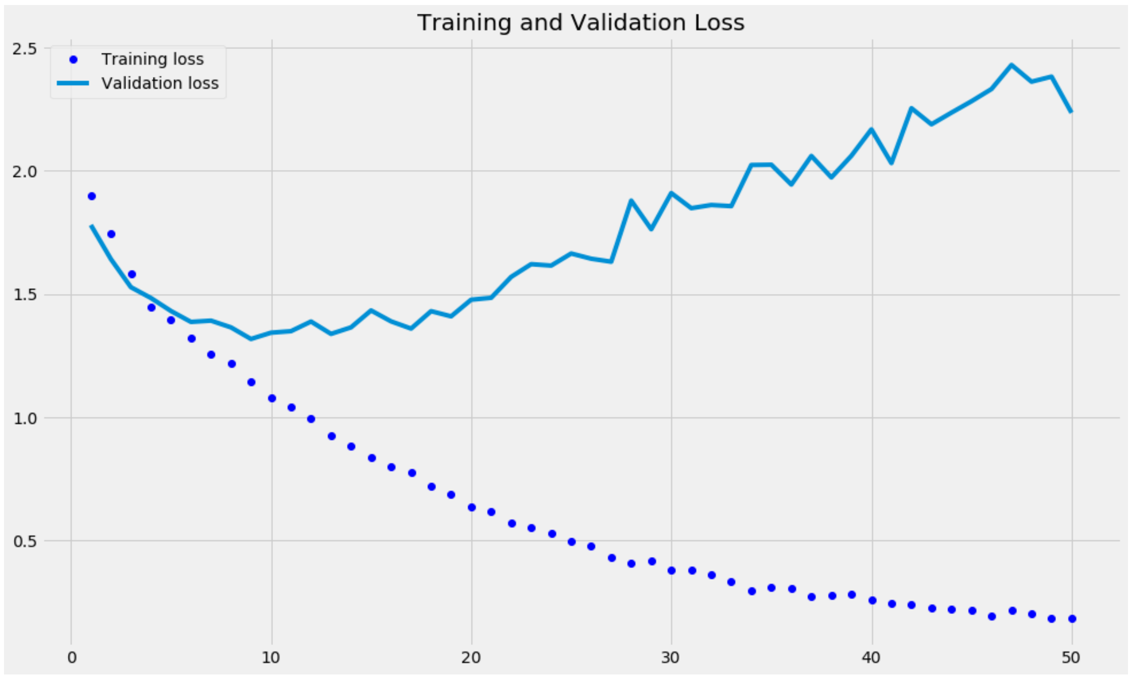
```
Epoch 40/50
563/563 [==============================] - 343s 609ms/step - loss: 0.2574 -
accuracy: 0.9086 - val_loss: 2.1683 - val_accuracy: 0.5579
Epoch 41/50
563/563 [==============================] - 340s 604ms/step - loss: 0.2476 -
accuracy: 0.9126 - val_loss: 2.0315 - val_accuracy: 0.5372
Epoch 42/50
563/563 [==============================] - 341s 606ms/step - loss: 0.2432 -
accuracy: 0.9146 - val_loss: 2.2542 - val_accuracy: 0.5248
Epoch 43/50
563/563 [==============================] - 341s 605ms/step - loss: 0.2271 -
accuracy: 0.9201 - val_loss: 2.1887 - val_accuracy: 0.5661
Epoch 44/50
563/563 [==============================] - 341s 605ms/step - loss: 0.2222 -
accuracy: 0.9218 - val_loss: 2.2362 - val_accuracy: 0.5413
Epoch 45/50
563/563 [==============================] - 341s 605ms/step - loss: 0.2175 -
accuracy: 0.9205 - val_loss: 2.2823 - val_accuracy: 0.5455
Epoch 46/50
563/563 [==============================] - 340s 604ms/step - loss: 0.1939 -
accuracy: 0.9322 - val_loss: 2.3320 - val_accuracy: 0.5455
Epoch 47/50
563/563 [==============================] - 341s 606ms/step - loss: 0.2161 -
accuracy: 0.9216 - val_loss: 2.4296 - val_accuracy: 0.5496
Epoch 48/50
563/563 [==============================] - 340s 604ms/step - loss: 0.2039 -
accuracy: 0.9247 - val_loss: 2.3617 - val_accuracy: 0.5661
Epoch 49/50
563/563 [==============================] - 340s 604ms/step - loss: 0.1820 -
accuracy: 0.9339 - val_loss: 2.3823 - val_accuracy: 0.5413
Epoch 50/50
563/563 [==============================] - 360s 639ms/step - loss: 0.1883 -
accuracy: 0.9322 - val_loss: 2.2355 - val_accuracy: 0.5496
```

[435]:
```
fig = plt.figure(figsize=(15, 20))

accuracy = history_4.history['accuracy']
val_accuracy = history_4.history['val_accuracy']
loss = history_4.history['loss']
val_loss = history_4.history['val_loss']

plt.subplot(2,1,1)
epochs = range(1, len(accuracy) + 1)
plt.plot(epochs, accuracy, 'bo', label='Training acc')
plt.plot(epochs, val_accuracy, label='Validation acc')
plt.title('Training and Validation Accuracy')
plt.legend()
```

```
fig1 = plt.figure()
fig1.savefig('train_val_acc_s.png')

fig = plt.figure(figsize=(15, 20))
plt.subplot(2,1,2)
plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, label='Validation loss')
plt.title('Training and Validation Loss')
plt.legend()
plt.figure()
plt.show();
fig.savefig('train_val_loss_s.png')
```



```
<Figure size 432x288 with 0 Axes>
```



```
<Figure size 432x288 with 0 Axes>
```

6. Key findings
   Accuracy is higher if more lower resolution images is used rather than small number of high resolution images.
   Loss keep increasing in every model and optimizer couldn't do anything to keep it in the right track.
   For low resolution images, accuracy is highest but unstable.
   Loss in low resolution images is 1 to 2.5 but in higher resolution images it is 2 to 6.
   For lower resolution images everything was done on unscaled, imbalanced class after fixing this issue only 800 picture left to analyse in case of higher resolution images.

```
[ ]:
```