In [1]:
```
# Convolutional Neural Network

# Installing Theano
# pip install --upgrade --no-deps git+git://github.com/Theano/Theano.git

# Installing Tensorflow
# Install Tensorflow from the website: https://www.tensorflow.org/versions/r0.
12/get_started/os_setup.html

# Installing Keras
# pip install --upgrade keras

# Part 1 - Building the CNN

# Importing the Keras libraries and packages
import numpy as np
import os
import keras_metrics
from keras.models import Sequential
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import TimeDistributed
from keras.layers import LSTM
from keras.layers import Reshape

import warnings
warnings.filterwarnings('ignore')

# Initialising the CNN
classifier = Sequential()

# Step 1 - Convolution
classifier.add(Convolution2D(64, (3, 3), padding = 'same', input_shape = (128,
 128, 3), activation = 'relu'))

# Step 2 - Pooling
classifier.add(MaxPooling2D(pool_size = (2, 2)))

# Adding a second convolutional layer
classifier.add(Convolution2D(64, (3, 3), padding = 'same', activation = 'relu'
))
classifier.add(MaxPooling2D(pool_size = (2, 2)))

# Adding a third conolutional layer
classifier.add(Convolution2D(64, (3, 3), padding = 'same', activation = 'relu'
))
classifier.add(MaxPooling2D(pool_size = (2, 2)))

# Step 3 - Flattening
classifier.add(Flatten())
classifier.add(Dropout(rate = 0.5))

# Step 4 - Full connection
```

```
classifier.add(Dense(output_dim = 128, activation = 'relu'))
classifier.add(Dropout(rate = 0.5))
classifier.add(Dense(output_dim = 7, activation = 'softmax'))

classifier.summary()
```

```
Z:\Anaconda3\lib\site-packages\h5py\__init__.py:36: FutureWarning: Conversion of the second argument of issubdtype from `float` to `np.floating` is deprecated. In future, it will be treated as `np.float64 == np.dtype(float).type`.
  from ._conv import register_converters as _register_converters
Using TensorFlow backend.

_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 128, 128, 64)      1792
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 64, 64, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 64, 64, 64)        36928
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 32, 32, 64)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 32, 32, 64)        36928
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 16, 16, 64)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 16384)             0
_________________________________________________________________
dropout_1 (Dropout)          (None, 16384)             0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               2097280
_________________________________________________________________
dropout_2 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_2 (Dense)              (None, 7)                 903
=================================================================
Total params: 2,173,831
Trainable params: 2,173,831
Non-trainable params: 0
_________________________________________________________________
```

In [2]:

```
# Compiling the CNN
classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy', keras_metrics.precision(), keras_metrics.recall()])
```

In [3]:
```
# Part 2 - Fitting the CNN to the images

from keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   height_shift_range =  0.1,
                                   width_shift_range = 0.1,
                                   channel_shift_range = 10)

test_datagen = ImageDataGenerator(rescale = 1./255)

training_set = train_datagen.flow_from_directory('train/',
                                                 target_size = (128, 128),
                                                 batch_size = 32,
                                                 class_mode = 'categorical')

test_set = test_datagen.flow_from_directory('test/',
                                            target_size = (128, 128),
                                            batch_size = 32,
                                            class_mode = 'categorical')
```

```
Found 4410 images belonging to 7 classes.
Found 1475 images belonging to 7 classes.
```

In [4]:
```
results = classifier.fit_generator(training_set,
                         samples_per_epoch = 4410,
                         nb_epoch = 100,
                         validation_data = test_set,
                         nb_val_samples = 1475)
```

```
Epoch 1/100
137/137 [==============================] - 993s 7s/step - loss: 1.5628 - acc:
0.3929 - precision: 0.5533 - recall: 0.1415 - val_loss: 1.2095 - val_acc: 0.5
665 - val_precision: 0.7853 - val_recall: 0.2206
Epoch 2/100
137/137 [==============================] - 1266s 9s/step - loss: 1.2770 - ac
c: 0.5051 - precision: 0.6919 - recall: 0.2861 - val_loss: 1.0006 - val_acc:
0.6136 - val_precision: 0.7737 - val_recall: 0.3900
Epoch 3/100
137/137 [==============================] - 1831s 13s/step - loss: 1.1648 - ac
c: 0.5520 - precision: 0.7100 - recall: 0.3584 - val_loss: 0.9996 - val_acc:
0.6202 - val_precision: 0.7643 - val_recall: 0.3972
Epoch 4/100
137/137 [==============================] - 1842s 13s/step - loss: 1.0854 - ac
c: 0.5749 - precision: 0.7181 - recall: 0.3954 - val_loss: 0.7919 - val_acc:
0.7193 - val_precision: 0.8122 - val_recall: 0.5350
Epoch 5/100
137/137 [==============================] - 1834s 13s/step - loss: 1.0322 - ac
c: 0.6031 - precision: 0.7268 - recall: 0.4306 - val_loss: 0.8408 - val_acc:
0.6862 - val_precision: 0.7572 - val_recall: 0.5555
Epoch 6/100
137/137 [==============================] - 1831s 13s/step - loss: 0.9928 - ac
c: 0.6052 - precision: 0.7278 - recall: 0.4482 - val_loss: 0.6830 - val_acc:
0.7664 - val_precision: 0.8284 - val_recall: 0.6355
Epoch 7/100
137/137 [==============================] - 1829s 13s/step - loss: 0.9311 - ac
c: 0.6385 - precision: 0.7366 - recall: 0.4946 - val_loss: 0.6259 - val_acc:
0.7706 - val_precision: 0.8500 - val_recall: 0.6745
Epoch 8/100
137/137 [==============================] - 1846s 13s/step - loss: 0.8966 - ac
c: 0.6524 - precision: 0.7515 - recall: 0.5258 - val_loss: 0.6064 - val_acc:
0.7807 - val_precision: 0.8615 - val_recall: 0.7008
Epoch 9/100
137/137 [==============================] - 1850s 14s/step - loss: 0.8613 - ac
c: 0.6698 - precision: 0.7572 - recall: 0.5427 - val_loss: 0.5174 - val_acc:
0.8174 - val_precision: 0.8649 - val_recall: 0.7658
Epoch 10/100
137/137 [==============================] - 1849s 13s/step - loss: 0.8137 - ac
c: 0.6851 - precision: 0.7670 - recall: 0.5757 - val_loss: 0.5954 - val_acc:
0.7586 - val_precision: 0.8229 - val_recall: 0.6929
Epoch 11/100
137/137 [==============================] - 1860s 14s/step - loss: 0.8313 - ac
c: 0.6821 - precision: 0.7609 - recall: 0.5623 - val_loss: 0.4963 - val_acc:
0.8127 - val_precision: 0.8716 - val_recall: 0.7518
Epoch 12/100
137/137 [==============================] - 1834s 13s/step - loss: 0.7708 - ac
c: 0.7062 - precision: 0.7819 - recall: 0.6027 - val_loss: 0.4024 - val_acc:
0.8697 - val_precision: 0.9059 - val_recall: 0.7956
Epoch 13/100
137/137 [==============================] - 1879s 14s/step - loss: 0.7324 - ac
c: 0.7144 - precision: 0.7802 - recall: 0.6262 - val_loss: 0.3662 - val_acc:
0.8704 - val_precision: 0.9020 - val_recall: 0.8367
Epoch 14/100
137/137 [==============================] - 1854s 14s/step - loss: 0.7525 - ac
c: 0.7150 - precision: 0.7818 - recall: 0.6143 - val_loss: 0.4394 - val_acc:
0.8488 - val_precision: 0.8949 - val_recall: 0.7926
Epoch 15/100
```

```
137/137 [==============================] - 1839s 13s/step - loss: 0.7074 - ac
c: 0.7273 - precision: 0.8069 - recall: 0.6438 - val_loss: 0.3612 - val_acc:
0.8658 - val_precision: 0.8963 - val_recall: 0.8203
Epoch 16/100
137/137 [==============================] - 1843s 13s/step - loss: 0.6650 - ac
c: 0.7406 - precision: 0.7971 - recall: 0.6647 - val_loss: 0.3041 - val_acc:
0.8888 - val_precision: 0.9130 - val_recall: 0.8684
Epoch 17/100
137/137 [==============================] - 1839s 13s/step - loss: 0.6458 - ac
c: 0.7455 - precision: 0.8057 - recall: 0.6788 - val_loss: 0.2857 - val_acc:
0.8982 - val_precision: 0.9134 - val_recall: 0.8724
Epoch 18/100
137/137 [==============================] - 1852s 14s/step - loss: 0.6596 - ac
c: 0.7425 - precision: 0.8055 - recall: 0.6769 - val_loss: 0.3170 - val_acc:
0.9024 - val_precision: 0.9438 - val_recall: 0.8644
Epoch 19/100
137/137 [==============================] - 1846s 13s/step - loss: 0.6174 - ac
c: 0.7584 - precision: 0.8117 - recall: 0.6949 - val_loss: 0.2950 - val_acc:
0.8976 - val_precision: 0.9187 - val_recall: 0.8718
Epoch 20/100
137/137 [==============================] - 1845s 13s/step - loss: 0.6245 - ac
c: 0.7597 - precision: 0.8115 - recall: 0.6943 - val_loss: 0.2457 - val_acc:
0.9086 - val_precision: 0.9244 - val_recall: 0.8937
Epoch 21/100
137/137 [==============================] - 1846s 13s/step - loss: 0.6116 - ac
c: 0.7554 - precision: 0.8093 - recall: 0.7010 - val_loss: 0.2783 - val_acc:
0.9191 - val_precision: 0.9481 - val_recall: 0.8831
Epoch 22/100
137/137 [==============================] - 1840s 13s/step - loss: 0.5690 - ac
c: 0.7748 - precision: 0.8231 - recall: 0.7254 - val_loss: 0.2455 - val_acc:
0.9235 - val_precision: 0.9395 - val_recall: 0.8922
Epoch 23/100
137/137 [==============================] - 1842s 13s/step - loss: 0.5622 - ac
c: 0.7764 - precision: 0.8283 - recall: 0.7237 - val_loss: 0.2486 - val_acc:
0.9213 - val_precision: 0.9404 - val_recall: 0.8976
Epoch 24/100
137/137 [==============================] - 1829s 13s/step - loss: 0.5844 - ac
c: 0.7788 - precision: 0.8269 - recall: 0.7225 - val_loss: 0.2040 - val_acc:
0.9377 - val_precision: 0.9577 - val_recall: 0.9214
Epoch 25/100
137/137 [==============================] - 1837s 13s/step - loss: 0.5524 - ac
c: 0.7851 - precision: 0.8332 - recall: 0.7386 - val_loss: 0.2087 - val_acc:
0.9390 - val_precision: 0.9489 - val_recall: 0.9172
Epoch 26/100
137/137 [==============================] - 1847s 13s/step - loss: 0.5296 - ac
c: 0.8049 - precision: 0.8484 - recall: 0.7560 - val_loss: 0.1894 - val_acc:
0.9485 - val_precision: 0.9563 - val_recall: 0.9322
Epoch 27/100
137/137 [==============================] - 1849s 13s/step - loss: 0.5140 - ac
c: 0.8024 - precision: 0.8470 - recall: 0.7590 - val_loss: 0.2195 - val_acc:
0.9321 - val_precision: 0.9442 - val_recall: 0.9085
Epoch 28/100
137/137 [==============================] - 1862s 14s/step - loss: 0.5159 - ac
c: 0.7996 - precision: 0.8408 - recall: 0.7545 - val_loss: 0.1993 - val_acc:
0.9220 - val_precision: 0.9365 - val_recall: 0.9098
Epoch 29/100
137/137 [==============================] - 1829s 13s/step - loss: 0.4758 - ac
```

```
c: 0.8172 - precision: 0.8520 - recall: 0.7802 - val_loss: 0.1845 - val_acc:
0.9498 - val_precision: 0.9623 - val_recall: 0.9348
Epoch 30/100
137/137 [==============================] - 1840s 13s/step - loss: 0.4909 - ac
c: 0.8159 - precision: 0.8519 - recall: 0.7757 - val_loss: 0.2116 - val_acc:
0.9281 - val_precision: 0.9430 - val_recall: 0.9091
Epoch 31/100
137/137 [==============================] - 1833s 13s/step - loss: 0.4823 - ac
c: 0.8077 - precision: 0.8475 - recall: 0.7664 - val_loss: 0.1374 - val_acc:
0.9592 - val_precision: 0.9633 - val_recall: 0.9457
Epoch 32/100
137/137 [==============================] - 1833s 13s/step - loss: 0.4921 - ac
c: 0.8140 - precision: 0.8511 - recall: 0.7714 - val_loss: 0.1876 - val_acc:
0.9452 - val_precision: 0.9590 - val_recall: 0.9343
Epoch 33/100
137/137 [==============================] - 1837s 13s/step - loss: 0.4763 - ac
c: 0.8139 - precision: 0.8492 - recall: 0.7771 - val_loss: 0.1725 - val_acc:
0.9404 - val_precision: 0.9488 - val_recall: 0.9295
Epoch 34/100
137/137 [==============================] - 1844s 13s/step - loss: 0.4543 - ac
c: 0.8300 - precision: 0.8616 - recall: 0.7904 - val_loss: 0.1274 - val_acc:
0.9591 - val_precision: 0.9641 - val_recall: 0.9517
Epoch 35/100
137/137 [==============================] - 1844s 13s/step - loss: 0.4583 - ac
c: 0.8238 - precision: 0.8555 - recall: 0.7873 - val_loss: 0.1422 - val_acc:
0.9559 - val_precision: 0.9622 - val_recall: 0.9484
Epoch 36/100
137/137 [==============================] - 1840s 13s/step - loss: 0.4286 - ac
c: 0.8388 - precision: 0.8688 - recall: 0.8096 - val_loss: 0.1453 - val_acc:
0.9579 - val_precision: 0.9641 - val_recall: 0.9505
Epoch 37/100
137/137 [==============================] - 1835s 13s/step - loss: 0.4610 - ac
c: 0.8296 - precision: 0.8604 - recall: 0.7947 - val_loss: 0.1219 - val_acc:
0.9641 - val_precision: 0.9689 - val_recall: 0.9525
Epoch 38/100
137/137 [==============================] - 1832s 13s/step - loss: 0.4229 - ac
c: 0.8409 - precision: 0.8710 - recall: 0.8081 - val_loss: 0.1069 - val_acc:
0.9688 - val_precision: 0.9780 - val_recall: 0.9613
Epoch 39/100
137/137 [==============================] - 1843s 13s/step - loss: 0.4381 - ac
c: 0.8296 - precision: 0.8623 - recall: 0.8006 - val_loss: 0.1082 - val_acc:
0.9709 - val_precision: 0.9741 - val_recall: 0.9661
Epoch 40/100
137/137 [==============================] - 1838s 13s/step - loss: 0.4066 - ac
c: 0.8456 - precision: 0.8738 - recall: 0.8195 - val_loss: 0.0965 - val_acc:
0.9750 - val_precision: 0.9802 - val_recall: 0.9688
Epoch 41/100
137/137 [==============================] - 1838s 13s/step - loss: 0.3972 - ac
c: 0.8460 - precision: 0.8737 - recall: 0.8205 - val_loss: 0.1012 - val_acc:
0.9668 - val_precision: 0.9713 - val_recall: 0.9628
Epoch 42/100
137/137 [==============================] - 1841s 13s/step - loss: 0.4016 - ac
c: 0.8489 - precision: 0.8708 - recall: 0.8204 - val_loss: 0.0959 - val_acc:
0.9763 - val_precision: 0.9803 - val_recall: 0.9736
Epoch 43/100
137/137 [==============================] - 1840s 13s/step - loss: 0.4187 - ac
c: 0.8436 - precision: 0.8707 - recall: 0.8104 - val_loss: 0.0904 - val_acc:
```

```
0.9783 - val_precision: 0.9815 - val_recall: 0.9729
Epoch 44/100
137/137 [==============================] - 1929s 14s/step - loss: 0.3989 - ac
c: 0.8504 - precision: 0.8804 - recall: 0.8205 - val_loss: 0.1351 - val_acc:
0.9525 - val_precision: 0.9574 - val_recall: 0.9457
Epoch 45/100
137/137 [==============================] - 1849s 13s/step - loss: 0.3986 - ac
c: 0.8524 - precision: 0.8760 - recall: 0.8245 - val_loss: 0.0982 - val_acc:
0.9702 - val_precision: 0.9721 - val_recall: 0.9668
Epoch 46/100
137/137 [==============================] - 1845s 13s/step - loss: 0.3921 - ac
c: 0.8526 - precision: 0.8809 - recall: 0.8255 - val_loss: 0.0914 - val_acc:
0.9688 - val_precision: 0.9726 - val_recall: 0.9627
Epoch 47/100
137/137 [==============================] - 1850s 14s/step - loss: 0.3903 - ac
c: 0.8495 - precision: 0.8778 - recall: 0.8253 - val_loss: 0.0883 - val_acc:
0.9721 - val_precision: 0.9774 - val_recall: 0.9694
Epoch 48/100
137/137 [==============================] - 1853s 14s/step - loss: 0.3890 - ac
c: 0.8565 - precision: 0.8800 - recall: 0.8348 - val_loss: 0.0785 - val_acc:
0.9783 - val_precision: 0.9822 - val_recall: 0.9708
Epoch 49/100
137/137 [==============================] - 1849s 13s/step - loss: 0.3845 - ac
c: 0.8560 - precision: 0.8803 - recall: 0.8310 - val_loss: 0.0875 - val_acc:
0.9735 - val_precision: 0.9801 - val_recall: 0.9702
Epoch 50/100
137/137 [==============================] - 1853s 14s/step - loss: 0.3961 - ac
c: 0.8518 - precision: 0.8753 - recall: 0.8235 - val_loss: 0.1179 - val_acc:
0.9520 - val_precision: 0.9576 - val_recall: 0.9466
Epoch 51/100
137/137 [==============================] - 1860s 14s/step - loss: 0.3862 - ac
c: 0.8579 - precision: 0.8841 - recall: 0.8314 - val_loss: 0.0745 - val_acc:
0.9823 - val_precision: 0.9843 - val_recall: 0.9816
Epoch 52/100
137/137 [==============================] - 1852s 14s/step - loss: 0.3670 - ac
c: 0.8600 - precision: 0.8814 - recall: 0.8375 - val_loss: 0.0731 - val_acc:
0.9791 - val_precision: 0.9810 - val_recall: 0.9729
Epoch 53/100
137/137 [==============================] - 1850s 14s/step - loss: 0.3722 - ac
c: 0.8620 - precision: 0.8824 - recall: 0.8333 - val_loss: 0.0807 - val_acc:
0.9782 - val_precision: 0.9815 - val_recall: 0.9735
Epoch 54/100
137/137 [==============================] - 1849s 13s/step - loss: 0.3593 - ac
c: 0.8670 - precision: 0.8898 - recall: 0.8440 - val_loss: 0.0657 - val_acc:
0.9817 - val_precision: 0.9843 - val_recall: 0.9790
Epoch 55/100
137/137 [==============================] - 1854s 14s/step - loss: 0.3166 - ac
c: 0.8824 - precision: 0.9013 - recall: 0.8609 - val_loss: 0.0626 - val_acc:
0.9825 - val_precision: 0.9838 - val_recall: 0.9798
Epoch 56/100
137/137 [==============================] - 1850s 14s/step - loss: 0.3454 - ac
c: 0.8751 - precision: 0.8934 - recall: 0.8550 - val_loss: 0.0783 - val_acc:
0.9784 - val_precision: 0.9830 - val_recall: 0.9757
Epoch 57/100
137/137 [==============================] - 1837s 13s/step - loss: 0.3511 - ac
c: 0.8672 - precision: 0.8874 - recall: 0.8437 - val_loss: 0.0744 - val_acc:
0.9776 - val_precision: 0.9809 - val_recall: 0.9756
```

```
Epoch 58/100
137/137 [==============================] - 1835s 13s/step - loss: 0.3382 - ac
c: 0.8714 - precision: 0.8921 - recall: 0.8540 - val_loss: 0.0694 - val_acc:
0.9796 - val_precision: 0.9816 - val_recall: 0.9776
Epoch 59/100
137/137 [==============================] - 1847s 13s/step - loss: 0.3088 - ac
c: 0.8890 - precision: 0.9029 - recall: 0.8685 - val_loss: 0.0648 - val_acc:
0.9823 - val_precision: 0.9830 - val_recall: 0.9810
Epoch 60/100
137/137 [==============================] - 1867s 14s/step - loss: 0.3464 - ac
c: 0.8770 - precision: 0.8957 - recall: 0.8572 - val_loss: 0.0818 - val_acc:
0.9750 - val_precision: 0.9808 - val_recall: 0.9695
Epoch 61/100
137/137 [==============================] - 1864s 14s/step - loss: 0.3166 - ac
c: 0.8800 - precision: 0.8959 - recall: 0.8631 - val_loss: 0.0647 - val_acc:
0.9850 - val_precision: 0.9903 - val_recall: 0.9803
Epoch 62/100
137/137 [==============================] - 1849s 13s/step - loss: 0.3161 - ac
c: 0.8771 - precision: 0.8941 - recall: 0.8631 - val_loss: 0.0732 - val_acc:
0.9804 - val_precision: 0.9843 - val_recall: 0.9763
Epoch 63/100
137/137 [==============================] - 1849s 13s/step - loss: 0.3326 - ac
c: 0.8748 - precision: 0.8954 - recall: 0.8553 - val_loss: 0.0640 - val_acc:
0.9898 - val_precision: 0.9905 - val_recall: 0.9851
Epoch 64/100
137/137 [==============================] - 1846s 13s/step - loss: 0.3421 - ac
c: 0.8722 - precision: 0.8923 - recall: 0.8528 - val_loss: 0.0645 - val_acc:
0.9891 - val_precision: 0.9891 - val_recall: 0.9844
Epoch 65/100
137/137 [==============================] - 1835s 13s/step - loss: 0.3161 - ac
c: 0.8804 - precision: 0.8979 - recall: 0.8568 - val_loss: 0.0696 - val_acc:
0.9803 - val_precision: 0.9836 - val_recall: 0.9776
Epoch 66/100
137/137 [==============================] - 1845s 13s/step - loss: 0.2990 - ac
c: 0.8865 - precision: 0.9012 - recall: 0.8719 - val_loss: 0.0623 - val_acc:
0.9844 - val_precision: 0.9864 - val_recall: 0.9823
Epoch 67/100
137/137 [==============================] - 1853s 14s/step - loss: 0.3316 - ac
c: 0.8757 - precision: 0.8982 - recall: 0.8515 - val_loss: 0.0534 - val_acc:
0.9878 - val_precision: 0.9891 - val_recall: 0.9844
Epoch 68/100
137/137 [==============================] - 1849s 13s/step - loss: 0.3215 - ac
c: 0.8843 - precision: 0.9017 - recall: 0.8663 - val_loss: 0.0644 - val_acc:
0.9838 - val_precision: 0.9858 - val_recall: 0.9817
Epoch 69/100
137/137 [==============================] - 1854s 14s/step - loss: 0.3186 - ac
c: 0.8841 - precision: 0.9019 - recall: 0.8686 - val_loss: 0.0690 - val_acc:
0.9803 - val_precision: 0.9809 - val_recall: 0.9770
Epoch 70/100
137/137 [==============================] - 1850s 14s/step - loss: 0.3057 - ac
c: 0.8885 - precision: 0.9044 - recall: 0.8714 - val_loss: 0.0539 - val_acc:
0.9837 - val_precision: 0.9850 - val_recall: 0.9810
Epoch 71/100
137/137 [==============================] - 1855s 14s/step - loss: 0.2995 - ac
c: 0.8885 - precision: 0.9065 - recall: 0.8722 - val_loss: 0.0572 - val_acc:
0.9824 - val_precision: 0.9830 - val_recall: 0.9790
Epoch 72/100
```

```
137/137 [==============================] - 1856s 14s/step - loss: 0.3089 - ac
c: 0.8816 - precision: 0.8969 - recall: 0.8653 - val_loss: 0.0648 - val_acc:
0.9830 - val_precision: 0.9850 - val_recall: 0.9823
Epoch 73/100
137/137 [==============================] - 1857s 14s/step - loss: 0.2994 - ac
c: 0.8876 - precision: 0.9056 - recall: 0.8686 - val_loss: 0.0364 - val_acc:
0.9932 - val_precision: 0.9939 - val_recall: 0.9919
Epoch 74/100
137/137 [==============================] - 1842s 13s/step - loss: 0.2925 - ac
c: 0.8971 - precision: 0.9078 - recall: 0.8796 - val_loss: 0.0574 - val_acc:
0.9864 - val_precision: 0.9891 - val_recall: 0.9844
Epoch 75/100
137/137 [==============================] - 1851s 14s/step - loss: 0.3014 - ac
c: 0.8926 - precision: 0.9105 - recall: 0.8749 - val_loss: 0.0376 - val_acc:
0.9926 - val_precision: 0.9959 - val_recall: 0.9913
Epoch 76/100
137/137 [==============================] - 1868s 14s/step - loss: 0.2748 - ac
c: 0.8982 - precision: 0.9141 - recall: 0.8847 - val_loss: 0.0484 - val_acc:
0.9865 - val_precision: 0.9878 - val_recall: 0.9844
Epoch 77/100
137/137 [==============================] - 1852s 14s/step - loss: 0.2952 - ac
c: 0.8992 - precision: 0.9151 - recall: 0.8840 - val_loss: 0.0428 - val_acc:
0.9946 - val_precision: 0.9952 - val_recall: 0.9919
Epoch 78/100
137/137 [==============================] - 1860s 14s/step - loss: 0.2999 - ac
c: 0.8859 - precision: 0.9030 - recall: 0.8711 - val_loss: 0.0413 - val_acc:
0.9946 - val_precision: 0.9946 - val_recall: 0.9933
Epoch 79/100
137/137 [==============================] - 1861s 14s/step - loss: 0.2707 - ac
c: 0.8941 - precision: 0.9092 - recall: 0.8788 - val_loss: 0.0449 - val_acc:
0.9912 - val_precision: 0.9932 - val_recall: 0.9898
Epoch 80/100
137/137 [==============================] - 1854s 14s/step - loss: 0.2842 - ac
c: 0.8964 - precision: 0.9121 - recall: 0.8807 - val_loss: 0.0302 - val_acc:
0.9939 - val_precision: 0.9953 - val_recall: 0.9919
Epoch 81/100
137/137 [==============================] - 1868s 14s/step - loss: 0.2727 - ac
c: 0.9025 - precision: 0.9179 - recall: 0.8874 - val_loss: 0.0315 - val_acc:
0.9926 - val_precision: 0.9932 - val_recall: 0.9905
Epoch 82/100
137/137 [==============================] - 1882s 14s/step - loss: 0.2918 - ac
c: 0.8928 - precision: 0.9059 - recall: 0.8755 - val_loss: 0.0279 - val_acc:
0.9946 - val_precision: 0.9966 - val_recall: 0.9946
Epoch 83/100
137/137 [==============================] - 1886s 14s/step - loss: 0.2660 - ac
c: 0.9044 - precision: 0.9139 - recall: 0.8911 - val_loss: 0.0407 - val_acc:
0.9878 - val_precision: 0.9898 - val_recall: 0.9865
Epoch 84/100
137/137 [==============================] - 1858s 14s/step - loss: 0.2755 - ac
c: 0.8998 - precision: 0.9145 - recall: 0.8847 - val_loss: 0.0325 - val_acc:
0.9905 - val_precision: 0.9925 - val_recall: 0.9878
Epoch 85/100
137/137 [==============================] - 1852s 14s/step - loss: 0.2845 - ac
c: 0.8984 - precision: 0.9093 - recall: 0.8832 - val_loss: 0.0494 - val_acc:
0.9844 - val_precision: 0.9884 - val_recall: 0.9837
Epoch 86/100
137/137 [==============================] - 1862s 14s/step - loss: 0.2728 - ac
```

```
c: 0.9011 - precision: 0.9149 - recall: 0.8867 - val_loss: 0.0458 - val_acc:
0.9844 - val_precision: 0.9890 - val_recall: 0.9824
Epoch 87/100
137/137 [==============================] - 1854s 14s/step - loss: 0.2667 - ac
c: 0.9036 - precision: 0.9178 - recall: 0.8906 - val_loss: 0.0341 - val_acc:
0.9960 - val_precision: 0.9966 - val_recall: 0.9960
Epoch 88/100
137/137 [==============================] - 1852s 14s/step - loss: 0.2757 - ac
c: 0.8990 - precision: 0.9127 - recall: 0.8855 - val_loss: 0.0288 - val_acc:
0.9945 - val_precision: 0.9952 - val_recall: 0.9925
Epoch 89/100
137/137 [==============================] - 1854s 14s/step - loss: 0.2618 - ac
c: 0.8994 - precision: 0.9118 - recall: 0.8863 - val_loss: 0.0287 - val_acc:
0.9953 - val_precision: 0.9966 - val_recall: 0.9946
Epoch 90/100
137/137 [==============================] - 1859s 14s/step - loss: 0.2699 - ac
c: 0.9021 - precision: 0.9139 - recall: 0.8866 - val_loss: 0.0294 - val_acc:
0.9959 - val_precision: 0.9959 - val_recall: 0.9932
Epoch 91/100
137/137 [==============================] - 1850s 14s/step - loss: 0.2615 - ac
c: 0.9036 - precision: 0.9138 - recall: 0.8895 - val_loss: 0.0368 - val_acc:
0.9925 - val_precision: 0.9939 - val_recall: 0.9912
Epoch 92/100
137/137 [==============================] - 1589s 12s/step - loss: 0.2612 - ac
c: 0.9031 - precision: 0.9153 - recall: 0.8933 - val_loss: 0.0340 - val_acc:
0.9912 - val_precision: 0.9912 - val_recall: 0.9912
Epoch 93/100
137/137 [==============================] - 1315s 10s/step - loss: 0.2473 - ac
c: 0.9112 - precision: 0.9215 - recall: 0.9001 - val_loss: 0.0346 - val_acc:
0.9919 - val_precision: 0.9939 - val_recall: 0.9912
Epoch 94/100
137/137 [==============================] - 975s 7s/step - loss: 0.2502 - acc:
0.9112 - precision: 0.9210 - recall: 0.8985 - val_loss: 0.0328 - val_acc: 0.9
878 - val_precision: 0.9885 - val_recall: 0.9878
Epoch 95/100
137/137 [==============================] - 975s 7s/step - loss: 0.2416 - acc:
0.9153 - precision: 0.9243 - recall: 0.9041 - val_loss: 0.0406 - val_acc: 0.9
870 - val_precision: 0.9884 - val_recall: 0.9857
Epoch 96/100
137/137 [==============================] - 975s 7s/step - loss: 0.2531 - acc:
0.9116 - precision: 0.9228 - recall: 0.8995 - val_loss: 0.0262 - val_acc: 0.9
939 - val_precision: 0.9952 - val_recall: 0.9932
Epoch 97/100
137/137 [==============================] - 978s 7s/step - loss: 0.2317 - acc:
0.9174 - precision: 0.9288 - recall: 0.9078 - val_loss: 0.0352 - val_acc: 0.9
878 - val_precision: 0.9891 - val_recall: 0.9865
Epoch 98/100
137/137 [==============================] - 976s 7s/step - loss: 0.2612 - acc:
0.9049 - precision: 0.9184 - recall: 0.8935 - val_loss: 0.0184 - val_acc: 0.9
980 - val_precision: 0.9980 - val_recall: 0.9966
Epoch 99/100
137/137 [==============================] - 977s 7s/step - loss: 0.2463 - acc:
0.9094 - precision: 0.9220 - recall: 0.8990 - val_loss: 0.0271 - val_acc: 0.9
939 - val_precision: 0.9946 - val_recall: 0.9926
Epoch 100/100
137/137 [==============================] - 977s 7s/step - loss: 0.2331 - acc:
```

```
0.9155 - precision: 0.9275 - recall: 0.9050 - val_loss: 0.0427 - val_acc: 0.9
878 - val_precision: 0.9884 - val_recall: 0.9844
```

In [5]:

```
test_steps_per_epoch = np.math.ceil(test_set.samples / test_set.batch_size)
predictions = classifier.predict_generator(test_set, steps=test_steps_per_epoc
h)
predicted_classes = np.argmax(predictions, axis=1)
```

In [6]:

```
true_classes = test_set.classes
class_labels = list(test_set.class_indices.keys())
```

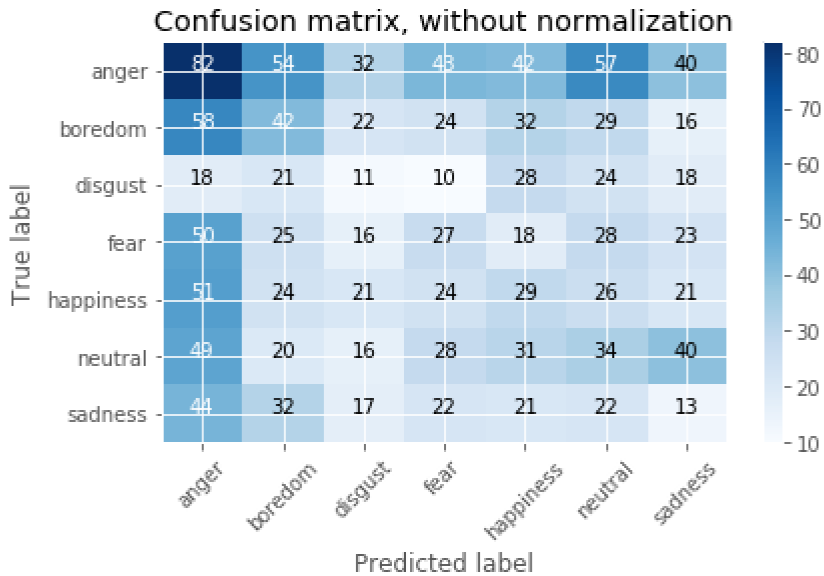
In [7]:

```
import sklearn.metrics as metrics
report = metrics.classification_report(true_classes, predicted_classes, target
_names=class_labels)
print(report)
```

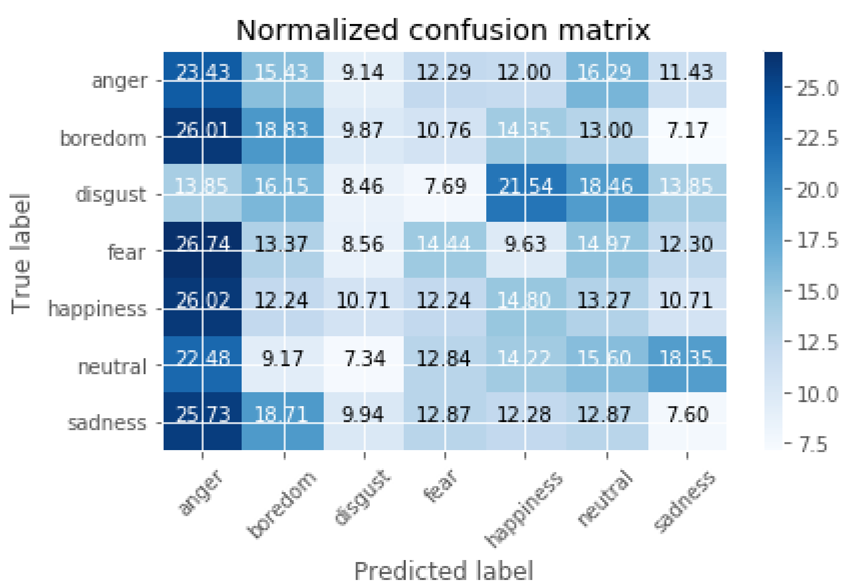
```
             precision    recall  f1-score   support

      anger       0.23      0.23      0.23       350
    boredom       0.19      0.19      0.19       223
    disgust       0.08      0.08      0.08       130
       fear       0.15      0.14      0.15       187
  happiness       0.14      0.15      0.15       196
    neutral       0.15      0.16      0.16       218
    sadness       0.08      0.08      0.08       171

avg / total       0.16      0.16      0.16      1475
```

In [10]:
```
import matplotlib.pyplot as plt
import itertools
def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):
    """
    This function prints and plots the confusion matrix.
    Normalization can be applied by setting normalize=True.
    """
    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]*100
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')

    print(cm)

    plt.imshow(cm, interpolation='nearest', cmap=cmap, aspect = 'auto')
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, format(cm[i, j], fmt),
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')

# Compute confusion matrix
cnf_matrix = metrics.confusion_matrix(true_classes, predicted_classes)
np.set_printoptions(precision=4)

# Plot non-normalized confusion matrix
plt.figure()
plot_confusion_matrix(cnf_matrix, classes=class_labels,
                      title='Confusion matrix, without normalization')
plt.savefig("non_normalized_confusion_matrix_cnn.png")
plt.show()
# Plot normalized confusion matrix
plt.figure()
plot_confusion_matrix(cnf_matrix, classes=class_labels, normalize=True,
                      title='Normalized confusion matrix')
plt.savefig("normalized_confusion_matrix_cnn.png")
plt.show()
```

```
Confusion matrix, without normalization
[[82 54 32 43 42 57 40]
 [58 42 22 24 32 29 16]
 [18 21 11 10 28 24 18]
 [50 25 16 27 18 28 23]
 [51 24 21 24 29 26 21]
 [49 20 16 28 31 34 40]
 [44 32 17 22 21 22 13]]
```




```
Normalized confusion matrix
[[23.4286 15.4286  9.1429 12.2857 12.     16.2857 11.4286]
 [26.009  18.8341  9.8655 10.7623 14.3498 13.0045  7.1749]
 [13.8462 16.1538  8.4615  7.6923 21.5385 18.4615 13.8462]
 [26.738  13.369   8.5561 14.4385  9.6257 14.9733 12.2995]
 [26.0204 12.2449 10.7143 12.2449 14.7959 13.2653 10.7143]
 [22.4771  9.1743  7.3394 12.844  14.2202 15.5963 18.3486]
 [25.731  18.7135  9.9415 12.8655 12.2807 12.8655  7.6023]]
```



In [11]:
```
import matplotlib.pyplot as plt
plt.style.use("ggplot")
plt.figure()
N = 100
plt.plot(np.arange(0, N), results.history["loss"], label="train_loss")
plt.plot(np.arange(0, N), results.history["val_loss"], label="val_loss")
plt.plot(np.arange(0, N), results.history["acc"], label="train_acc")
plt.plot(np.arange(0, N), results.history["val_acc"], label="val_acc")
plt.title("Training Loss and Accuracy")
plt.xlabel("Epoch #")
plt.ylabel("Loss/Accuracy")
plt.legend(loc="upper left")
plt.savefig("plot_cnn.png")
```