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(Reshape((4*4, 1024)))
classifier.add(LSTM(units = 50, return_sequences = True, dropout = 0.5))
classifier.add(LSTM(units = 20, return_sequences = False, dropout = 0.5))
classifier.add(Dense(output_dim = 8, 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 depreca
ted. 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
reshape_1 (Reshape)	(None, 16, 1024)	0
lstm_1 (LSTM)	(None, 16, 50)	215000
lstm_2 (LSTM)	(None, 20)	5680
dense_1 (Dense)	(None, 8)	168

```
Total params: 296,496
Trainable params: 296,496
Non-trainable params: 0
```

In [2]:

```
# Compiling the CNN
classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metr
ics = ['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 11880 images belonging to 8 classes.
Found 3960 images belonging to 8 classes.
```

In [4]:
```
results = classifier.fit_generator(training_set,
                         samples_per_epoch = 11880,
                         nb_epoch = 100,
                         validation_data = test_set,
                         nb_val_samples = 3960)
```

```
Epoch 1/100
371/371 [==============================] - 2670s 7s/step - loss: 1.9911 - ac
c: 0.2082 - precision: 0.0000e+00 - recall: 0.0000e+00 - val_loss: 1.8958 - v
al_acc: 0.2358 - val_precision: 0.0000e+00 - val_recall: 0.0000e+00
Epoch 2/100
371/371 [==============================] - 2667s 7s/step - loss: 1.8522 - ac
c: 0.2806 - precision: 0.1188 - recall: 0.0047 - val_loss: 1.6662 - val_acc:
0.3714 - val_precision: 0.6508 - val_recall: 0.0487
Epoch 3/100
371/371 [==============================] - 2658s 7s/step - loss: 1.7524 - ac
c: 0.3286 - precision: 0.5181 - recall: 0.0386 - val_loss: 1.5928 - val_acc:
0.3996 - val_precision: 0.6663 - val_recall: 0.0897
Epoch 4/100
371/371 [==============================] - 2662s 7s/step - loss: 1.7023 - ac
c: 0.3507 - precision: 0.5503 - recall: 0.0613 - val_loss: 1.5128 - val_acc:
0.4359 - val_precision: 0.7229 - val_recall: 0.1422
Epoch 5/100
371/371 [==============================] - 2669s 7s/step - loss: 1.6347 - ac
c: 0.3817 - precision: 0.6172 - recall: 0.1065 - val_loss: 1.4503 - val_acc:
0.4439 - val_precision: 0.6817 - val_recall: 0.2260
Epoch 6/100
371/371 [==============================] - 2664s 7s/step - loss: 1.5804 - ac
c: 0.4062 - precision: 0.6256 - recall: 0.1494 - val_loss: 1.3575 - val_acc:
0.5008 - val_precision: 0.7080 - val_recall: 0.2429
Epoch 7/100
371/371 [==============================] - 2658s 7s/step - loss: 1.5349 - ac
c: 0.4263 - precision: 0.6393 - recall: 0.1845 - val_loss: 1.2864 - val_acc:
0.5237 - val_precision: 0.7183 - val_recall: 0.3095
Epoch 8/100
371/371 [==============================] - 2655s 7s/step - loss: 1.4606 - ac
c: 0.4549 - precision: 0.6554 - recall: 0.2257 - val_loss: 1.2898 - val_acc:
0.5265 - val_precision: 0.7026 - val_recall: 0.3149
Epoch 9/100
371/371 [==============================] - 2652s 7s/step - loss: 1.4223 - ac
c: 0.4752 - precision: 0.6654 - recall: 0.2512 - val_loss: 1.2727 - val_acc:
0.5273 - val_precision: 0.6915 - val_recall: 0.3268
Epoch 10/100
371/371 [==============================] - 2661s 7s/step - loss: 1.3741 - ac
c: 0.4906 - precision: 0.6828 - recall: 0.2837 - val_loss: 1.1433 - val_acc:
0.5800 - val_precision: 0.7657 - val_recall: 0.3843
Epoch 11/100
371/371 [==============================] - 2655s 7s/step - loss: 1.3353 - ac
c: 0.4980 - precision: 0.6894 - recall: 0.3005 - val_loss: 1.1008 - val_acc:
0.6081 - val_precision: 0.7498 - val_recall: 0.4056
Epoch 12/100
371/371 [==============================] - 2657s 7s/step - loss: 1.3042 - ac
c: 0.5190 - precision: 0.6901 - recall: 0.3191 - val_loss: 1.0338 - val_acc:
0.6241 - val_precision: 0.7704 - val_recall: 0.4506
Epoch 13/100
371/371 [==============================] - 2659s 7s/step - loss: 1.2485 - ac
c: 0.5386 - precision: 0.7068 - recall: 0.3522 - val_loss: 0.9604 - val_acc:
0.6449 - val_precision: 0.7755 - val_recall: 0.4997
Epoch 14/100
371/371 [==============================] - 2659s 7s/step - loss: 1.2120 - ac
c: 0.5513 - precision: 0.7125 - recall: 0.3716 - val_loss: 1.0064 - val_acc:
0.6372 - val_precision: 0.7588 - val_recall: 0.4991
Epoch 15/100
```

```
371/371 [==============================] - 2647s 7s/step - loss: 1.1961 - ac
c: 0.5637 - precision: 0.7158 - recall: 0.3854 - val_loss: 0.9004 - val_acc:
0.6754 - val_precision: 0.8100 - val_recall: 0.5311
Epoch 16/100
371/371 [==============================] - 2651s 7s/step - loss: 1.1560 - ac
c: 0.5697 - precision: 0.7179 - recall: 0.4089 - val_loss: 0.9148 - val_acc:
0.6705 - val_precision: 0.7990 - val_recall: 0.5225
Epoch 17/100
371/371 [==============================] - 2653s 7s/step - loss: 1.1338 - ac
c: 0.5820 - precision: 0.7294 - recall: 0.4180 - val_loss: 0.8941 - val_acc:
0.6710 - val_precision: 0.7918 - val_recall: 0.5488
Epoch 18/100
371/371 [==============================] - 2644s 7s/step - loss: 1.0965 - ac
c: 0.5956 - precision: 0.7328 - recall: 0.4434 - val_loss: 0.8375 - val_acc:
0.6957 - val_precision: 0.8076 - val_recall: 0.5770
Epoch 19/100
371/371 [==============================] - 2648s 7s/step - loss: 1.0845 - ac
c: 0.6028 - precision: 0.7252 - recall: 0.4474 - val_loss: 0.8249 - val_acc:
0.6959 - val_precision: 0.7998 - val_recall: 0.5875
Epoch 20/100
371/371 [==============================] - 2651s 7s/step - loss: 1.0691 - ac
c: 0.6071 - precision: 0.7361 - recall: 0.4625 - val_loss: 0.8181 - val_acc:
0.6992 - val_precision: 0.7862 - val_recall: 0.5962
Epoch 21/100
371/371 [==============================] - 2718s 7s/step - loss: 1.0544 - ac
c: 0.6106 - precision: 0.7370 - recall: 0.4666 - val_loss: 0.7737 - val_acc:
0.7157 - val_precision: 0.8043 - val_recall: 0.6290
Epoch 22/100
371/371 [==============================] - 2776s 7s/step - loss: 1.0344 - ac
c: 0.6219 - precision: 0.7387 - recall: 0.4870 - val_loss: 0.7216 - val_acc:
0.7359 - val_precision: 0.8366 - val_recall: 0.6339
Epoch 23/100
371/371 [==============================] - 2733s 7s/step - loss: 1.0193 - ac
c: 0.6218 - precision: 0.7381 - recall: 0.4859 - val_loss: 0.7657 - val_acc:
0.7137 - val_precision: 0.8047 - val_recall: 0.6167
Epoch 24/100
371/371 [==============================] - 2715s 7s/step - loss: 1.0238 - ac
c: 0.6256 - precision: 0.7421 - recall: 0.4955 - val_loss: 0.7382 - val_acc:
0.7231 - val_precision: 0.8015 - val_recall: 0.6472
Epoch 25/100
371/371 [==============================] - 2694s 7s/step - loss: 0.9967 - ac
c: 0.6314 - precision: 0.7472 - recall: 0.5085 - val_loss: 0.6982 - val_acc:
0.7435 - val_precision: 0.8241 - val_recall: 0.6601
Epoch 26/100
371/371 [==============================] - 2665s 7s/step - loss: 0.9736 - ac
c: 0.6440 - precision: 0.7477 - recall: 0.5222 - val_loss: 0.6512 - val_acc:
0.7672 - val_precision: 0.8601 - val_recall: 0.6702
Epoch 27/100
371/371 [==============================] - 2662s 7s/step - loss: 0.9585 - ac
c: 0.6483 - precision: 0.7504 - recall: 0.5350 - val_loss: 0.6796 - val_acc:
0.7545 - val_precision: 0.8215 - val_recall: 0.6679
Epoch 28/100
371/371 [==============================] - 2645s 7s/step - loss: 0.9222 - ac
c: 0.6634 - precision: 0.7678 - recall: 0.5497 - val_loss: 0.6054 - val_acc:
0.7851 - val_precision: 0.8521 - val_recall: 0.7040
Epoch 29/100
371/371 [==============================] - 2646s 7s/step - loss: 0.9256 - ac
```

```
c: 0.6605 - precision: 0.7590 - recall: 0.5467 - val_loss: 0.5856 - val_acc:
0.8013 - val_precision: 0.8615 - val_recall: 0.7112
Epoch 30/100
371/371 [==============================] - 2639s 7s/step - loss: 0.9122 - ac
c: 0.6660 - precision: 0.7627 - recall: 0.5588 - val_loss: 0.5722 - val_acc:
0.7950 - val_precision: 0.8555 - val_recall: 0.7205
Epoch 31/100
371/371 [==============================] - 2641s 7s/step - loss: 0.8827 - ac
c: 0.6787 - precision: 0.7763 - recall: 0.5741 - val_loss: 0.5908 - val_acc:
0.7846 - val_precision: 0.8381 - val_recall: 0.7190
Epoch 32/100
371/371 [==============================] - 2646s 7s/step - loss: 0.8880 - ac
c: 0.6754 - precision: 0.7639 - recall: 0.5744 - val_loss: 0.5576 - val_acc:
0.8063 - val_precision: 0.8669 - val_recall: 0.7303
Epoch 33/100
371/371 [==============================] - 2638s 7s/step - loss: 0.8613 - ac
c: 0.6851 - precision: 0.7728 - recall: 0.5835 - val_loss: 0.5804 - val_acc:
0.7834 - val_precision: 0.8382 - val_recall: 0.7210
Epoch 34/100
371/371 [==============================] - 2692s 7s/step - loss: 0.8619 - ac
c: 0.6849 - precision: 0.7748 - recall: 0.5895 - val_loss: 0.5643 - val_acc:
0.7957 - val_precision: 0.8583 - val_recall: 0.7285
Epoch 35/100
371/371 [==============================] - 2669s 7s/step - loss: 0.8488 - ac
c: 0.6862 - precision: 0.7727 - recall: 0.5994 - val_loss: 0.4923 - val_acc:
0.8293 - val_precision: 0.8803 - val_recall: 0.7724
Epoch 36/100
371/371 [==============================] - 2664s 7s/step - loss: 0.8293 - ac
c: 0.6974 - precision: 0.7789 - recall: 0.6055 - val_loss: 0.5439 - val_acc:
0.8093 - val_precision: 0.8683 - val_recall: 0.7472
Epoch 37/100
371/371 [==============================] - 2666s 7s/step - loss: 0.8159 - ac
c: 0.7034 - precision: 0.7811 - recall: 0.6140 - val_loss: 0.5009 - val_acc:
0.8189 - val_precision: 0.8723 - val_recall: 0.7636
Epoch 38/100
371/371 [==============================] - 2668s 7s/step - loss: 0.7986 - ac
c: 0.7095 - precision: 0.7929 - recall: 0.6276 - val_loss: 0.4711 - val_acc:
0.8316 - val_precision: 0.8720 - val_recall: 0.7811
Epoch 39/100
371/371 [==============================] - 2672s 7s/step - loss: 0.8031 - ac
c: 0.7110 - precision: 0.7841 - recall: 0.6258 - val_loss: 0.4444 - val_acc:
0.8505 - val_precision: 0.8976 - val_recall: 0.8058
Epoch 40/100
371/371 [==============================] - 2661s 7s/step - loss: 0.8051 - ac
c: 0.7106 - precision: 0.7842 - recall: 0.6262 - val_loss: 0.5255 - val_acc:
0.8159 - val_precision: 0.8649 - val_recall: 0.7601
Epoch 41/100
371/371 [==============================] - 2671s 7s/step - loss: 0.7788 - ac
c: 0.7179 - precision: 0.7920 - recall: 0.6377 - val_loss: 0.4430 - val_acc:
0.8477 - val_precision: 0.8878 - val_recall: 0.7949
Epoch 42/100
371/371 [==============================] - 2674s 7s/step - loss: 0.7719 - ac
c: 0.7172 - precision: 0.7898 - recall: 0.6386 - val_loss: 0.4100 - val_acc:
0.8616 - val_precision: 0.9047 - val_recall: 0.8217
Epoch 43/100
371/371 [==============================] - 2677s 7s/step - loss: 0.7632 - ac
c: 0.7270 - precision: 0.7974 - recall: 0.6472 - val_loss: 0.4818 - val_acc:
```

```
0.8335 - val_precision: 0.8715 - val_recall: 0.7934
Epoch 44/100
371/371 [==============================] - 2668s 7s/step - loss: 0.7626 - ac
c: 0.7264 - precision: 0.7959 - recall: 0.6528 - val_loss: 0.4663 - val_acc:
0.8332 - val_precision: 0.8733 - val_recall: 0.7973
Epoch 45/100
371/371 [==============================] - 2665s 7s/step - loss: 0.7694 - ac
c: 0.7182 - precision: 0.7875 - recall: 0.6441 - val_loss: 0.4603 - val_acc:
0.8439 - val_precision: 0.8846 - val_recall: 0.7954
Epoch 46/100
371/371 [==============================] - 2665s 7s/step - loss: 0.7215 - ac
c: 0.7428 - precision: 0.8059 - recall: 0.6693 - val_loss: 0.3989 - val_acc:
0.8598 - val_precision: 0.9021 - val_recall: 0.8236
Epoch 47/100
371/371 [==============================] - 2673s 7s/step - loss: 0.7142 - ac
c: 0.7414 - precision: 0.8046 - recall: 0.6723 - val_loss: 0.4164 - val_acc:
0.8472 - val_precision: 0.8867 - val_recall: 0.8106
Epoch 48/100
371/371 [==============================] - 2671s 7s/step - loss: 0.7182 - ac
c: 0.7460 - precision: 0.8092 - recall: 0.6752 - val_loss: 0.3828 - val_acc:
0.8699 - val_precision: 0.9013 - val_recall: 0.8293
Epoch 49/100
371/371 [==============================] - 2671s 7s/step - loss: 0.7087 - ac
c: 0.7475 - precision: 0.8090 - recall: 0.6778 - val_loss: 0.3757 - val_acc:
0.8728 - val_precision: 0.8993 - val_recall: 0.8349
Epoch 50/100
371/371 [==============================] - 2669s 7s/step - loss: 0.7225 - ac
c: 0.7416 - precision: 0.8045 - recall: 0.6712 - val_loss: 0.3686 - val_acc:
0.8755 - val_precision: 0.9083 - val_recall: 0.8422
Epoch 51/100
371/371 [==============================] - 2664s 7s/step - loss: 0.6903 - ac
c: 0.7536 - precision: 0.8151 - recall: 0.6856 - val_loss: 0.4221 - val_acc:
0.8507 - val_precision: 0.8771 - val_recall: 0.8245
Epoch 52/100
371/371 [==============================] - 2662s 7s/step - loss: 0.7002 - ac
c: 0.7516 - precision: 0.8119 - recall: 0.6888 - val_loss: 0.4660 - val_acc:
0.8288 - val_precision: 0.8637 - val_recall: 0.7970
Epoch 53/100
371/371 [==============================] - 2665s 7s/step - loss: 0.6894 - ac
c: 0.7531 - precision: 0.8117 - recall: 0.6869 - val_loss: 0.3474 - val_acc:
0.8858 - val_precision: 0.9127 - val_recall: 0.8553
Epoch 54/100
371/371 [==============================] - 2668s 7s/step - loss: 0.6720 - ac
c: 0.7601 - precision: 0.8174 - recall: 0.6964 - val_loss: 0.4196 - val_acc:
0.8583 - val_precision: 0.8886 - val_recall: 0.8288
Epoch 55/100
371/371 [==============================] - 2666s 7s/step - loss: 0.6782 - ac
c: 0.7587 - precision: 0.8152 - recall: 0.6977 - val_loss: 0.3367 - val_acc:
0.8825 - val_precision: 0.9120 - val_recall: 0.8578
Epoch 56/100
371/371 [==============================] - 2663s 7s/step - loss: 0.6774 - ac
c: 0.7551 - precision: 0.8134 - recall: 0.6986 - val_loss: 0.3325 - val_acc:
0.8883 - val_precision: 0.9177 - val_recall: 0.8641
Epoch 57/100
371/371 [==============================] - 2679s 7s/step - loss: 0.6559 - ac
c: 0.7668 - precision: 0.8242 - recall: 0.7055 - val_loss: 0.4184 - val_acc:
0.8516 - val_precision: 0.8762 - val_recall: 0.8278
```

```
Epoch 58/100
371/371 [==============================] - 2680s 7s/step - loss: 0.6669 - ac
c: 0.7636 - precision: 0.8182 - recall: 0.7073 - val_loss: 0.3445 - val_acc:
0.8825 - val_precision: 0.9037 - val_recall: 0.8575
Epoch 59/100
371/371 [==============================] - 2678s 7s/step - loss: 0.6466 - ac
c: 0.7657 - precision: 0.8178 - recall: 0.7093 - val_loss: 0.3016 - val_acc:
0.8980 - val_precision: 0.9183 - val_recall: 0.8803
Epoch 60/100
371/371 [==============================] - 2662s 7s/step - loss: 0.6425 - ac
c: 0.7700 - precision: 0.8230 - recall: 0.7178 - val_loss: 0.3611 - val_acc:
0.8720 - val_precision: 0.8962 - val_recall: 0.8503
Epoch 61/100
371/371 [==============================] - 2668s 7s/step - loss: 0.6528 - ac
c: 0.7635 - precision: 0.8215 - recall: 0.7132 - val_loss: 0.2889 - val_acc:
0.9080 - val_precision: 0.9293 - val_recall: 0.8871
Epoch 62/100
371/371 [==============================] - 2642s 7s/step - loss: 0.6363 - ac
c: 0.7732 - precision: 0.8268 - recall: 0.7191 - val_loss: 0.2999 - val_acc:
0.9015 - val_precision: 0.9220 - val_recall: 0.8770
Epoch 63/100
371/371 [==============================] - 2645s 7s/step - loss: 0.6351 - ac
c: 0.7749 - precision: 0.8240 - recall: 0.7186 - val_loss: 0.2917 - val_acc:
0.9006 - val_precision: 0.9257 - val_recall: 0.8768
Epoch 64/100
371/371 [==============================] - 2649s 7s/step - loss: 0.6294 - ac
c: 0.7754 - precision: 0.8281 - recall: 0.7229 - val_loss: 0.2877 - val_acc:
0.9046 - val_precision: 0.9247 - val_recall: 0.8839
Epoch 65/100
371/371 [==============================] - 2639s 7s/step - loss: 0.6204 - ac
c: 0.7758 - precision: 0.8271 - recall: 0.7246 - val_loss: 0.2914 - val_acc:
0.9033 - val_precision: 0.9229 - val_recall: 0.8823
Epoch 66/100
371/371 [==============================] - 2637s 7s/step - loss: 0.6175 - ac
c: 0.7785 - precision: 0.8281 - recall: 0.7303 - val_loss: 0.2847 - val_acc:
0.9061 - val_precision: 0.9234 - val_recall: 0.8798
Epoch 67/100
371/371 [==============================] - 2645s 7s/step - loss: 0.6143 - ac
c: 0.7812 - precision: 0.8313 - recall: 0.7282 - val_loss: 0.2922 - val_acc:
0.8989 - val_precision: 0.9195 - val_recall: 0.8800
Epoch 68/100
371/371 [==============================] - 2646s 7s/step - loss: 0.6071 - ac
c: 0.7827 - precision: 0.8326 - recall: 0.7340 - val_loss: 0.3095 - val_acc:
0.8917 - val_precision: 0.9134 - val_recall: 0.8735
Epoch 69/100
371/371 [==============================] - 2651s 7s/step - loss: 0.5989 - ac
c: 0.7850 - precision: 0.8336 - recall: 0.7348 - val_loss: 0.2782 - val_acc:
0.9017 - val_precision: 0.9203 - val_recall: 0.8869
Epoch 70/100
371/371 [==============================] - 2644s 7s/step - loss: 0.6064 - ac
c: 0.7867 - precision: 0.8330 - recall: 0.7354 - val_loss: 0.2837 - val_acc:
0.9023 - val_precision: 0.9239 - val_recall: 0.8869
Epoch 71/100
371/371 [==============================] - 2649s 7s/step - loss: 0.5899 - ac
c: 0.7902 - precision: 0.8391 - recall: 0.7412 - val_loss: 0.3112 - val_acc:
0.8937 - val_precision: 0.9127 - val_recall: 0.8762
Epoch 72/100
```

```
371/371 [==============================] - 2646s 7s/step - loss: 0.5942 - ac
c: 0.7882 - precision: 0.8365 - recall: 0.7423 - val_loss: 0.2508 - val_acc:
0.9180 - val_precision: 0.9328 - val_recall: 0.8973
Epoch 73/100
371/371 [==============================] - 2643s 7s/step - loss: 0.5868 - ac
c: 0.7937 - precision: 0.8397 - recall: 0.7485 - val_loss: 0.2546 - val_acc:
0.9124 - val_precision: 0.9299 - val_recall: 0.8944
Epoch 74/100
371/371 [==============================] - 2647s 7s/step - loss: 0.6006 - ac
c: 0.7883 - precision: 0.8371 - recall: 0.7439 - val_loss: 0.2555 - val_acc:
0.9104 - val_precision: 0.9303 - val_recall: 0.8953
Epoch 75/100
371/371 [==============================] - 2657s 7s/step - loss: 0.5910 - ac
c: 0.7918 - precision: 0.8401 - recall: 0.7473 - val_loss: 0.2571 - val_acc:
0.9149 - val_precision: 0.9302 - val_recall: 0.8950
Epoch 76/100
371/371 [==============================] - 2643s 7s/step - loss: 0.5816 - ac
c: 0.7969 - precision: 0.8414 - recall: 0.7490 - val_loss: 0.2613 - val_acc:
0.9085 - val_precision: 0.9290 - val_recall: 0.8929
Epoch 77/100
371/371 [==============================] - 2647s 7s/step - loss: 0.5734 - ac
c: 0.7989 - precision: 0.8429 - recall: 0.7536 - val_loss: 0.2425 - val_acc:
0.9197 - val_precision: 0.9380 - val_recall: 0.9048
Epoch 78/100
371/371 [==============================] - 2649s 7s/step - loss: 0.5573 - ac
c: 0.8040 - precision: 0.8486 - recall: 0.7580 - val_loss: 0.2166 - val_acc:
0.9280 - val_precision: 0.9434 - val_recall: 0.9134
Epoch 79/100
371/371 [==============================] - 2650s 7s/step - loss: 0.5744 - ac
c: 0.7993 - precision: 0.8432 - recall: 0.7541 - val_loss: 0.2472 - val_acc:
0.9220 - val_precision: 0.9354 - val_recall: 0.9063
Epoch 80/100
371/371 [==============================] - 2647s 7s/step - loss: 0.5682 - ac
c: 0.8037 - precision: 0.8400 - recall: 0.7620 - val_loss: 0.2310 - val_acc:
0.9287 - val_precision: 0.9434 - val_recall: 0.9090
Epoch 81/100
371/371 [==============================] - 2649s 7s/step - loss: 0.5454 - ac
c: 0.8060 - precision: 0.8494 - recall: 0.7645 - val_loss: 0.2527 - val_acc:
0.9170 - val_precision: 0.9341 - val_recall: 0.8975
Epoch 82/100
371/371 [==============================] - 2647s 7s/step - loss: 0.5693 - ac
c: 0.8026 - precision: 0.8462 - recall: 0.7623 - val_loss: 0.2836 - val_acc:
0.9035 - val_precision: 0.9224 - val_recall: 0.8858
Epoch 83/100
371/371 [==============================] - 2652s 7s/step - loss: 0.5437 - ac
c: 0.8048 - precision: 0.8490 - recall: 0.7644 - val_loss: 0.2626 - val_acc:
0.9111 - val_precision: 0.9287 - val_recall: 0.8979
Epoch 84/100
371/371 [==============================] - 2659s 7s/step - loss: 0.5336 - ac
c: 0.8133 - precision: 0.8536 - recall: 0.7686 - val_loss: 0.2419 - val_acc:
0.9204 - val_precision: 0.9337 - val_recall: 0.9055
Epoch 85/100
371/371 [==============================] - 2642s 7s/step - loss: 0.5532 - ac
c: 0.8004 - precision: 0.8413 - recall: 0.7570 - val_loss: 0.2470 - val_acc:
0.9184 - val_precision: 0.9352 - val_recall: 0.9047
Epoch 86/100
371/371 [==============================] - 2652s 7s/step - loss: 0.5453 - ac
```

```
c: 0.8083 - precision: 0.8503 - recall: 0.7656 - val_loss: 0.2454 - val_acc:
0.9174 - val_precision: 0.9302 - val_recall: 0.9018
Epoch 87/100
371/371 [==============================] - 2668s 7s/step - loss: 0.5232 - ac
c: 0.8160 - precision: 0.8562 - recall: 0.7785 - val_loss: 0.2238 - val_acc:
0.9259 - val_precision: 0.9377 - val_recall: 0.9121
Epoch 88/100
371/371 [==============================] - 2661s 7s/step - loss: 0.5432 - ac
c: 0.8091 - precision: 0.8529 - recall: 0.7681 - val_loss: 0.2325 - val_acc:
0.9187 - val_precision: 0.9352 - val_recall: 0.9040
Epoch 89/100
371/371 [==============================] - 2694s 7s/step - loss: 0.5340 - ac
c: 0.8115 - precision: 0.8526 - recall: 0.7723 - val_loss: 0.2044 - val_acc:
0.9326 - val_precision: 0.9448 - val_recall: 0.9238
Epoch 90/100
371/371 [==============================] - 2686s 7s/step - loss: 0.5257 - ac
c: 0.8122 - precision: 0.8507 - recall: 0.7730 - val_loss: 0.2157 - val_acc:
0.9291 - val_precision: 0.9409 - val_recall: 0.9157
Epoch 91/100
371/371 [==============================] - 2699s 7s/step - loss: 0.5420 - ac
c: 0.8103 - precision: 0.8509 - recall: 0.7748 - val_loss: 0.2566 - val_acc:
0.9080 - val_precision: 0.9241 - val_recall: 0.8949
Epoch 92/100
371/371 [==============================] - 2685s 7s/step - loss: 0.5318 - ac
c: 0.8151 - precision: 0.8544 - recall: 0.7733 - val_loss: 0.2435 - val_acc:
0.9230 - val_precision: 0.9350 - val_recall: 0.9064
Epoch 93/100
371/371 [==============================] - 2648s 7s/step - loss: 0.5356 - ac
c: 0.8138 - precision: 0.8525 - recall: 0.7779 - val_loss: 0.2037 - val_acc:
0.9341 - val_precision: 0.9478 - val_recall: 0.9212
Epoch 94/100
371/371 [==============================] - 2662s 7s/step - loss: 0.5214 - ac
c: 0.8170 - precision: 0.8540 - recall: 0.7811 - val_loss: 0.2309 - val_acc:
0.9225 - val_precision: 0.9380 - val_recall: 0.9087
Epoch 95/100
371/371 [==============================] - 2720s 7s/step - loss: 0.5173 - ac
c: 0.8170 - precision: 0.8557 - recall: 0.7811 - val_loss: 0.2030 - val_acc:
0.9326 - val_precision: 0.9441 - val_recall: 0.9210
Epoch 96/100
371/371 [==============================] - 2704s 7s/step - loss: 0.5123 - ac
c: 0.8197 - precision: 0.8569 - recall: 0.7808 - val_loss: 0.1956 - val_acc:
0.9358 - val_precision: 0.9490 - val_recall: 0.9262
Epoch 97/100
371/371 [==============================] - 2692s 7s/step - loss: 0.5292 - ac
c: 0.8137 - precision: 0.8498 - recall: 0.7771 - val_loss: 0.2430 - val_acc:
0.9217 - val_precision: 0.9337 - val_recall: 0.9101
Epoch 98/100
371/371 [==============================] - 2657s 7s/step - loss: 0.5019 - ac
c: 0.8245 - precision: 0.8604 - recall: 0.7880 - val_loss: 0.2188 - val_acc:
0.9238 - val_precision: 0.9357 - val_recall: 0.9099
Epoch 99/100
371/371 [==============================] - 2550s 7s/step - loss: 0.5224 - ac
c: 0.8164 - precision: 0.8552 - recall: 0.7820 - val_loss: 0.2282 - val_acc:
0.9242 - val_precision: 0.9348 - val_recall: 0.9131
Epoch 100/100
371/371 [==============================] - 1543s 4s/step - loss: 0.5096 - ac
```

```
c: 0.8221 - precision: 0.8589 - recall: 0.7844 - val_loss: 0.1977 - val_acc:
0.9354 - val_precision: 0.9498 - val_recall: 0.9251
```

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)
```

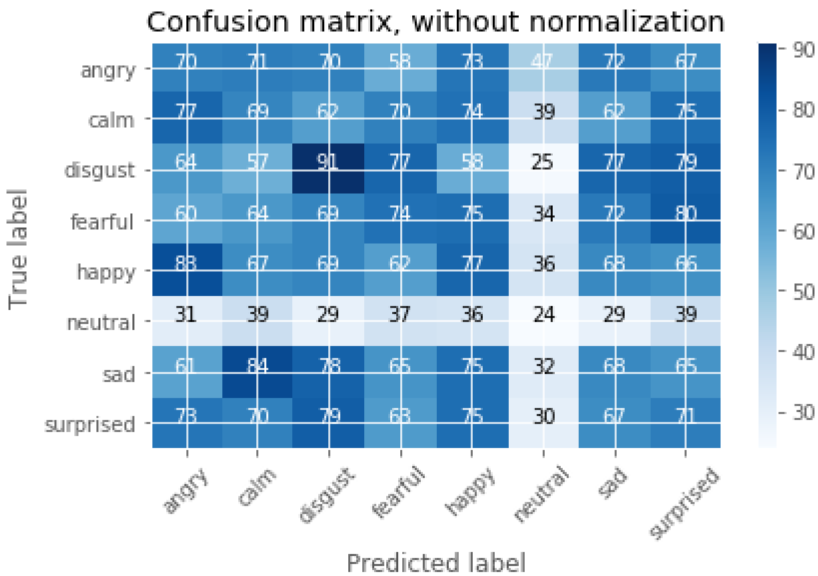
In [6]:
```
true_classes = test_set.classes
class_labels = list(test_set.class_indices.keys())
```

In [7]:
```
import sklearn.metrics as metrics
report = metrics.classification_report(true_classes, predicted_classes, target
_names=class_labels)
print(report)
```

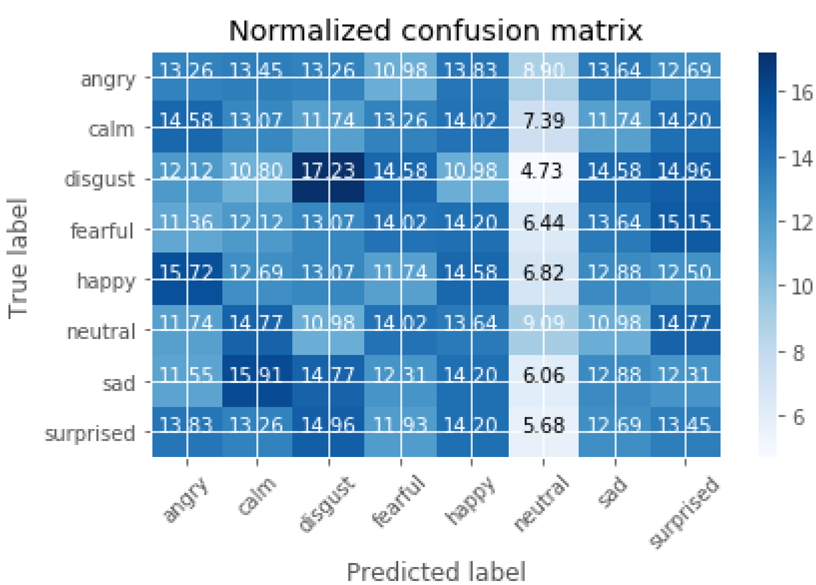
```
             precision    recall  f1-score   support

      angry       0.13      0.13      0.13       528
       calm       0.13      0.13      0.13       528
    disgust       0.17      0.17      0.17       528
    fearful       0.15      0.14      0.14       528
      happy       0.14      0.15      0.14       528
    neutral       0.09      0.09      0.09       264
        sad       0.13      0.13      0.13       528
  surprised       0.13      0.13      0.13       528

avg / total       0.14      0.14      0.14      3960
```

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_lstm.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_lstm.png")
plt.show()
```

```
Confusion matrix, without normalization
[[70 71 70 58 73 47 72 67]
 [77 69 62 70 74 39 62 75]
 [64 57 91 77 58 25 77 79]
 [60 64 69 74 75 34 72 80]
 [83 67 69 62 77 36 68 66]
 [31 39 29 37 36 24 29 39]
 [61 84 78 65 75 32 68 65]
 [73 70 79 63 75 30 67 71]]
```




```
Normalized confusion matrix
[[13.2576 13.447  13.2576 10.9848 13.8258  8.9015 13.6364 12.6894]
 [14.5833 13.0682 11.7424 13.2576 14.0152  7.3864 11.7424 14.2045]
 [12.1212 10.7955 17.2348 14.5833 10.9848  4.7348 14.5833 14.9621]
 [11.3636 12.1212 13.0682 14.0152 14.2045  6.4394 13.6364 15.1515]
 [15.7197 12.6894 13.0682 11.7424 14.5833  6.8182 12.8788 12.5   ]
 [11.7424 14.7727 10.9848 14.0152 13.6364  9.0909 10.9848 14.7727]
 [11.553  15.9091 14.7727 12.3106 14.2045  6.0606 12.8788 12.3106]
 [13.8258 13.2576 14.9621 11.9318 14.2045  5.6818 12.6894 13.447 ]]
```



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_lstm.png")
```