AudioProject

February 19, 2019

0.1 1. Importing needed libraries

In [1]:
```
import librosa
import librosa.display


import keras
from keras import regularizers
from keras.models import Sequential, load_model
from keras.layers import Conv2D, MaxPooling2D, Activation, Flatten, Dropout, Dense, Gl
from keras.callbacks import ModelCheckpoint, EarlyStopping


import random
import numpy as np
from sklearn.model_selection import train_test_split
import pandas as pd
import matplotlib.pyplot as plt

import fnmatch
import os
import sys


import warnings
warnings.filterwarnings('ignore')

%matplotlib inline
```

```
Using TensorFlow backend.
```

In [2]:
```
plt.rcParams['figure.figsize'] = [15, 10]  # Make output graphs larger in Jupyter Note
```

GPU activation for Nvidia GPUs only

In [3]:
```
import tensorflow as tf
sess = tf.Session(config=tf.ConfigProto(log_device_placement=True))
```

```
In [4]: # Check that there is an output
        from keras import backend as k
        k.tensorflow_backend._get_available_gpus()
```

```
Out[4]: []
```

0.2 2. Data Preparation

```
In [4]: data = pd.read_csv('data/metadata.csv')
        data.head(10)
```

```
Out[4]:       slice_file_name    fsID      start        end  salience  fold  classID  \
        0    100032-3-0-0.wav  100032   0.000000   0.317551         1     5        3
        1  100263-2-0-117.wav  100263  58.500000  62.500000         1     5        2
        2  100263-2-0-121.wav  100263  60.500000  64.500000         1     5        2
        3  100263-2-0-126.wav  100263  63.000000  67.000000         1     5        2
        4  100263-2-0-137.wav  100263  68.500000  72.500000         1     5        2
        5  100263-2-0-143.wav  100263  71.500000  75.500000         1     5        2
        6  100263-2-0-161.wav  100263  80.500000  84.500000         1     5        2
        7    100263-2-0-3.wav  100263   1.500000   5.500000         1     5        2
        8   100263-2-0-36.wav  100263  18.000000  22.000000         1     5        2
        9    100648-1-0-0.wav  100648   4.823402   5.471927         2    10        1

                        class
        0           dog_bark
        1  children_playing
        2  children_playing
        3  children_playing
        4  children_playing
        5  children_playing
        6  children_playing
        7  children_playing
        8  children_playing
        9           car_horn
```

```
In [5]: print("The original data containes {0} audio files.".format(data.shape[0]))

        # Filter data less than 3 seconds
        dataorg = data[['slice_file_name', 'fold' ,'classID', 'class']]
        data3s = data[['slice_file_name', 'fold' ,'classID', 'class']][ data['end']-data['start

        print("The data we will work with containes {0} audio files of 3 seconds length.".forma

        data3s.head(10)
```

```
The original data containes 8732 audio files.
The data we will work with containes 7468 audio files of 3 seconds length.
```

```
Out[5]:        slice_file_name  fold  classID            class
        1   100263-2-0-117.wav     5        2  children_playing
        2   100263-2-0-121.wav     5        2  children_playing
        3   100263-2-0-126.wav     5        2  children_playing
        4   100263-2-0-137.wav     5        2  children_playing
        5   100263-2-0-143.wav     5        2  children_playing
        6   100263-2-0-161.wav     5        2  children_playing
        7     100263-2-0-3.wav     5        2  children_playing
        8    100263-2-0-36.wav     5        2  children_playing
        14    100652-3-0-0.wav     2        3         dog_bark
        15    100652-3-0-1.wav     2        3         dog_bark
```

Different Classes are: 0. air_conditioner
1. car_horn
2. children_playing
3. dog_bark
4. drilling
5. engine_idling 6. gun_shot 7. jackhammer 8. siren 9. street_music

Iterate over all samples in *valid*. For every sample, construct the (128,128) spectrogram

```
In [6]: data3s['path'] = 'fold' + data3s['fold'].astype('str') + '/' + data3s['slice_file_name
```

```
In [7]: dataset = []

        for row in data3s.itertuples():
            y, sr = librosa.load('data/UrbanSound8K/' + row.path, duration=2.97)
            ps = librosa.feature.melspectrogram(y=y, sr=sr)
            if ps.shape != (128, 128): continue
            dataset.append( (ps, row.classID) )
```

```
In [15]: random.shuffle(dataset)
         #X_train, X_test, y_train, y_test = train_test_split(dataset, test_size = .2)

         train = dataset[:7000]
         test = dataset[7000:]

         X_train, y_train = zip(*train)
         X_test, y_test = zip(*test)

         # Reshape for CNN input
         X_train = np.array([x.reshape( (128, 128, 1) ) for x in X_train])
         X_test = np.array([x.reshape( (128, 128, 1) ) for x in X_test])

         # One-Hot encoding for classes
         y_train = np.array(keras.utils.to_categorical(y_train, 10))
         y_test = np.array(keras.utils.to_categorical(y_test, 10))
```

```
In [31]: model = keras.models.Sequential()
         input_shape=(128, 128, 1)
```

```
        # model.add(Conv2D(96, (3, 3), activation='relu', padding = 'same', input_shape=input_
        # model.add(Dropout(0.2))
        # model.add(Conv2D(96, (3, 3), activation='relu', padding = 'same'))
        # model.add(Conv2D(96, (3, 3), activation='relu', padding = 'same', strides = 2))
        # model.add(Dropout(0.5))
        # model.add(Conv2D(192, (3, 3), activation='relu', padding = 'same'))
        # model.add(Conv2D(192, (3, 3), activation='relu', padding = 'same'))
        # model.add(Conv2D(256, (3, 3), activation='relu', padding = 'same', strides = 2))
        # model.add(Conv2D(512, (3, 3), activation='relu', padding = 'same', strides = 2))
        # model.add(Conv2D(512, (3, 3), activation='relu', padding = 'same', strides = 2))
        # model.add(Dropout(0.5))
        # model.add(Activation('relu'))


        model.add(Conv2D(24, (5, 5), strides=(1, 1), input_shape=input_shape))
        model.add(MaxPooling2D((4, 2), strides=(4, 2)))
        model.add(Activation('relu'))

        model.add(Conv2D(48, (5, 5), padding="valid"))
        model.add(MaxPooling2D((4, 2), strides=(4, 2)))
        model.add(Activation('relu'))

        model.add(Conv2D(48, (5, 5), padding="valid"))
        model.add(Activation('relu'))

        model.add(Flatten())
        model.add(Dropout(rate=0.5))

        model.add(Dense(64))
        model.add(Activation('relu'))
        model.add(Dropout(rate=0.5))

        model.add(Dense(10))
        model.add(Activation('softmax'))

In [32]: model.compile(
             optimizer="Adam",
             loss="categorical_crossentropy",
             metrics=['accuracy'])

In [34]: best_model = ModelCheckpoint('best_model.hdf5', save_best_only=True, monitor='val_loss

         history = model.fit(
                 x=X_train,
                 y=y_train,
```

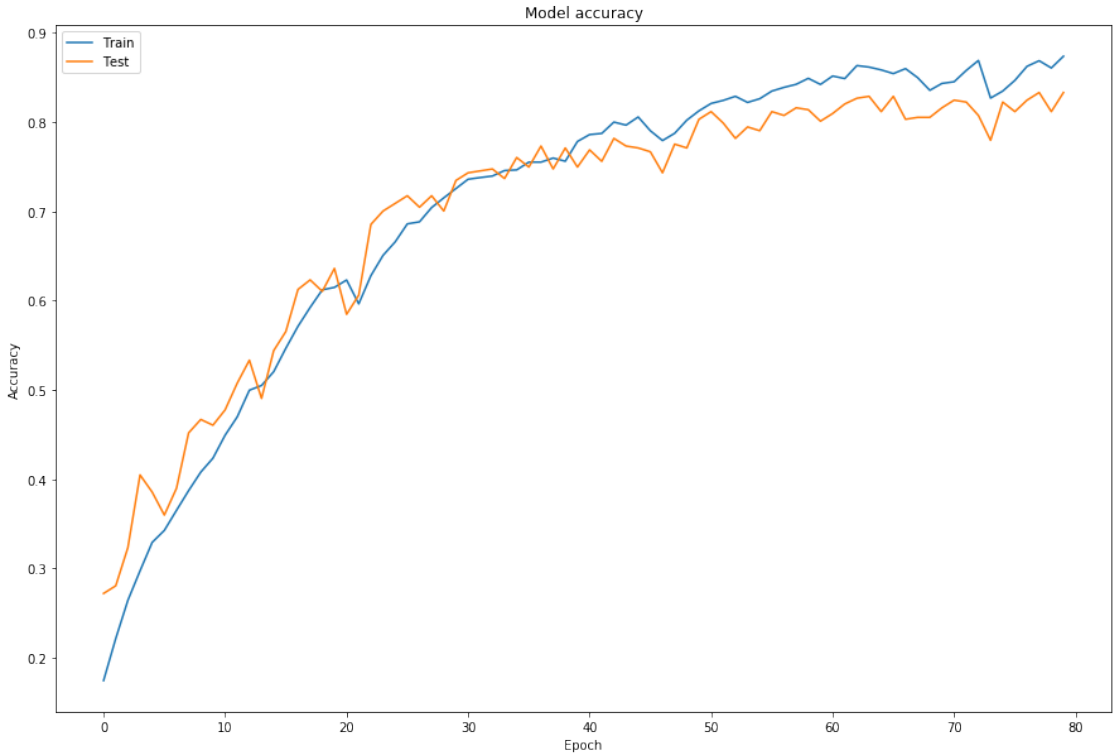
```
                epochs=80,
                batch_size=500,
                validation_data= (X_test, y_test),
                callbacks=[best_model] )

        plt.plot(history.history['acc'])
        plt.plot(history.history['val_acc'])
        plt.title('Model accuracy')
        plt.ylabel('Accuracy')
        plt.xlabel('Epoch')
        plt.legend(['Train', 'Test'], loc='upper left')
        plt.show()

        # Plot training & validation loss values
        plt.plot(history.history['loss'])
        plt.plot(history.history['val_loss'])
        plt.title('Model loss')
        plt.ylabel('Loss')
        plt.xlabel('Epoch')
        plt.legend(['Train', 'Test'], loc='upper left')
        plt.show()

        print('Test loss:', score[0])
        print('Test accuracy:', score[1])
```

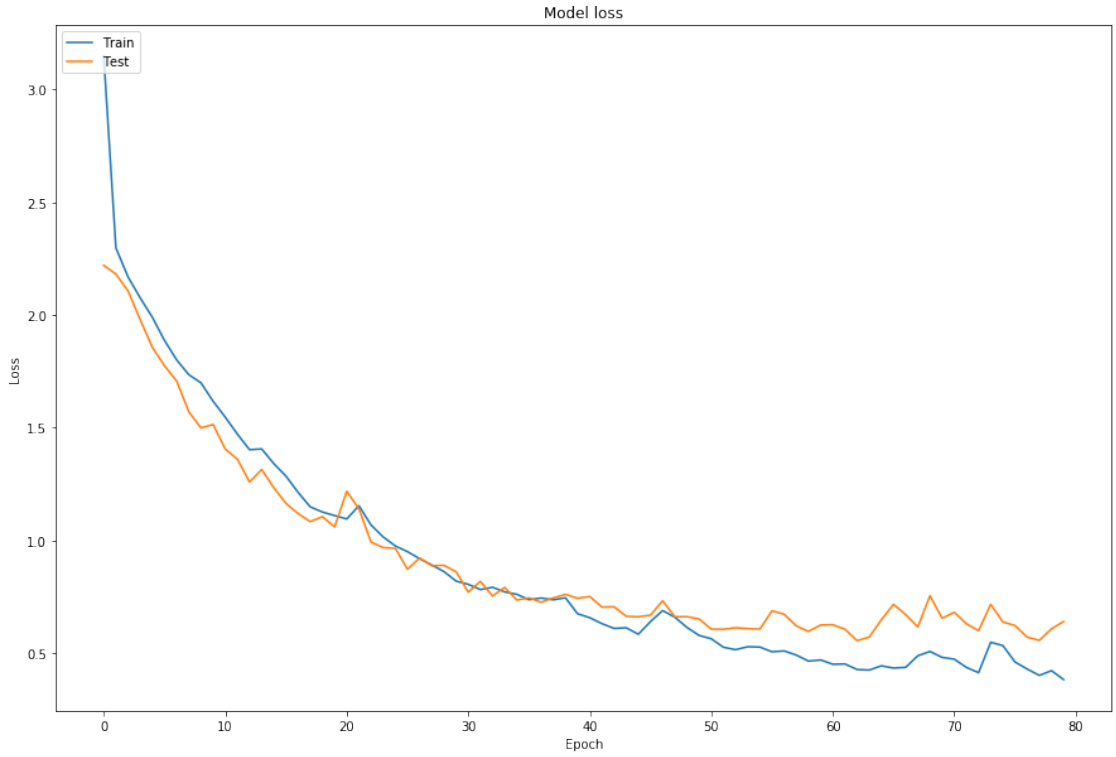
```
Train on 7000 samples, validate on 467 samples
Epoch 1/80
7000/7000 [==============================] - 64s 9ms/step - loss: 3.1472 - acc: 0.1744 - val_lo
Epoch 2/80
7000/7000 [==============================] - 62s 9ms/step - loss: 2.2972 - acc: 0.2214 - val_lo
Epoch 3/80
7000/7000 [==============================] - 63s 9ms/step - loss: 2.1687 - acc: 0.2643 - val_lo
Epoch 4/80
7000/7000 [==============================] - 65s 9ms/step - loss: 2.0756 - acc: 0.2973 - val_lo
Epoch 5/80
7000/7000 [==============================] - 62s 9ms/step - loss: 1.9899 - acc: 0.3291 - val_lo
Epoch 6/80
7000/7000 [==============================] - 63s 9ms/step - loss: 1.8871 - acc: 0.3426 - val_lo
Epoch 7/80
7000/7000 [==============================] - 64s 9ms/step - loss: 1.8010 - acc: 0.3650 - val_lo
Epoch 8/80
7000/7000 [==============================] - 65s 9ms/step - loss: 1.7353 - acc: 0.3871 - val_lo
Epoch 9/80
7000/7000 [==============================] - 63s 9ms/step - loss: 1.6994 - acc: 0.4079 - val_lo
Epoch 10/80
7000/7000 [==============================] - 63s 9ms/step - loss: 1.6176 - acc: 0.4234 - val_lo
Epoch 11/80
7000/7000 [==============================] - 61s 9ms/step - loss: 1.5471 - acc: 0.4491 - val_lo
```

```
Epoch 12/80
7000/7000 [==============================] - 61s 9ms/step - loss: 1.4719 - acc: 0.4697 - val_lo
Epoch 13/80
7000/7000 [==============================] - 63s 9ms/step - loss: 1.4027 - acc: 0.4996 - val_lo
Epoch 14/80
7000/7000 [==============================] - 63s 9ms/step - loss: 1.4062 - acc: 0.5049 - val_lo
Epoch 15/80
7000/7000 [==============================] - 61s 9ms/step - loss: 1.3408 - acc: 0.5203 - val_lo
Epoch 16/80
7000/7000 [==============================] - 62s 9ms/step - loss: 1.2855 - acc: 0.5466 - val_lo
Epoch 17/80
7000/7000 [==============================] - 63s 9ms/step - loss: 1.2133 - acc: 0.5711 - val_lo
Epoch 18/80
7000/7000 [==============================] - 61s 9ms/step - loss: 1.1491 - acc: 0.5923 - val_lo
Epoch 19/80
7000/7000 [==============================] - 62s 9ms/step - loss: 1.1265 - acc: 0.6120 - val_lo
Epoch 20/80
7000/7000 [==============================] - 63s 9ms/step - loss: 1.1108 - acc: 0.6147 - val_lo
Epoch 21/80
7000/7000 [==============================] - 63s 9ms/step - loss: 1.0957 - acc: 0.6230 - val_lo
Epoch 22/80
7000/7000 [==============================] - 62s 9ms/step - loss: 1.1544 - acc: 0.5963 - val_lo
Epoch 23/80
7000/7000 [==============================] - 64s 9ms/step - loss: 1.0688 - acc: 0.6279 - val_lo
Epoch 24/80
7000/7000 [==============================] - 63s 9ms/step - loss: 1.0161 - acc: 0.6506 - val_lo
Epoch 25/80
7000/7000 [==============================] - 62s 9ms/step - loss: 0.9760 - acc: 0.6657 - val_lo
Epoch 26/80
7000/7000 [==============================] - 63s 9ms/step - loss: 0.9507 - acc: 0.6859 - val_lo
Epoch 27/80
7000/7000 [==============================] - 64s 9ms/step - loss: 0.9194 - acc: 0.6881 - val_lo
Epoch 28/80
7000/7000 [==============================] - 61s 9ms/step - loss: 0.8916 - acc: 0.7041 - val_lo
Epoch 29/80
7000/7000 [==============================] - 61s 9ms/step - loss: 0.8624 - acc: 0.7150 - val_lo
Epoch 30/80
7000/7000 [==============================] - 63s 9ms/step - loss: 0.8203 - acc: 0.7254 - val_lo
Epoch 31/80
7000/7000 [==============================] - 61s 9ms/step - loss: 0.8061 - acc: 0.7359 - val_lo
Epoch 32/80
7000/7000 [==============================] - 63s 9ms/step - loss: 0.7826 - acc: 0.7377 - val_lo
Epoch 33/80
7000/7000 [==============================] - 63s 9ms/step - loss: 0.7930 - acc: 0.7394 - val_lo
Epoch 34/80
7000/7000 [==============================] - 62s 9ms/step - loss: 0.7724 - acc: 0.7456 - val_lo
Epoch 35/80
7000/7000 [==============================] - 62s 9ms/step - loss: 0.7616 - acc: 0.7463 - val_lo
```

```
Epoch 36/80
7000/7000 [==============================] - 62s 9ms/step - loss: 0.7378 - acc: 0.7550 - val_lo
Epoch 37/80
7000/7000 [==============================] - 62s 9ms/step - loss: 0.7452 - acc: 0.7550 - val_lo
Epoch 38/80
7000/7000 [==============================] - 63s 9ms/step - loss: 0.7373 - acc: 0.7596 - val_lo
Epoch 39/80
7000/7000 [==============================] - 62s 9ms/step - loss: 0.7463 - acc: 0.7560 - val_lo
Epoch 40/80
7000/7000 [==============================] - 61s 9ms/step - loss: 0.6755 - acc: 0.7781 - val_lo
Epoch 41/80
7000/7000 [==============================] - 63s 9ms/step - loss: 0.6580 - acc: 0.7859 - val_lo
Epoch 42/80
7000/7000 [==============================] - 62s 9ms/step - loss: 0.6318 - acc: 0.7871 - val_lo
Epoch 43/80
7000/7000 [==============================] - 65s 9ms/step - loss: 0.6105 - acc: 0.7999 - val_lo
Epoch 44/80
7000/7000 [==============================] - 65s 9ms/step - loss: 0.6134 - acc: 0.7964 - val_lo
Epoch 45/80
7000/7000 [==============================] - 63s 9ms/step - loss: 0.5847 - acc: 0.8056 - val_lo
Epoch 46/80
7000/7000 [==============================] - 62s 9ms/step - loss: 0.6409 - acc: 0.7901 - val_lo
Epoch 47/80
7000/7000 [==============================] - 64s 9ms/step - loss: 0.6897 - acc: 0.7791 - val_lo
Epoch 48/80
7000/7000 [==============================] - 62s 9ms/step - loss: 0.6602 - acc: 0.7873 - val_lo
Epoch 49/80
7000/7000 [==============================] - 60s 9ms/step - loss: 0.6149 - acc: 0.8020 - val_lo
Epoch 50/80
7000/7000 [==============================] - 61s 9ms/step - loss: 0.5794 - acc: 0.8124 - val_lo
Epoch 51/80
7000/7000 [==============================] - 60s 9ms/step - loss: 0.5647 - acc: 0.8207 - val_lo
Epoch 52/80
7000/7000 [==============================] - 60s 9ms/step - loss: 0.5271 - acc: 0.8241 - val_lo
Epoch 53/80
7000/7000 [==============================] - 60s 9ms/step - loss: 0.5163 - acc: 0.8287 - val_lo
Epoch 54/80
7000/7000 [==============================] - 61s 9ms/step - loss: 0.5293 - acc: 0.8219 - val_lo
Epoch 55/80
7000/7000 [==============================] - 61s 9ms/step - loss: 0.5279 - acc: 0.8260 - val_lo
Epoch 56/80
7000/7000 [==============================] - 63s 9ms/step - loss: 0.5070 - acc: 0.8346 - val_lo
Epoch 57/80
7000/7000 [==============================] - 62s 9ms/step - loss: 0.5107 - acc: 0.8387 - val_lo
Epoch 58/80
7000/7000 [==============================] - 67s 10ms/step - loss: 0.4924 - acc: 0.8421 - val_l
Epoch 59/80
7000/7000 [==============================] - 62s 9ms/step - loss: 0.4655 - acc: 0.8489 - val_lo
```

```
Epoch 60/80
7000/7000 [==============================] - 62s 9ms/step - loss: 0.4708 - acc: 0.8420 - val_lo
Epoch 61/80
7000/7000 [==============================] - 63s 9ms/step - loss: 0.4517 - acc: 0.8514 - val_lo
Epoch 62/80
7000/7000 [==============================] - 63s 9ms/step - loss: 0.4530 - acc: 0.8486 - val_lo
Epoch 63/80
7000/7000 [==============================] - 63s 9ms/step - loss: 0.4285 - acc: 0.8631 - val_lo
Epoch 64/80
7000/7000 [==============================] - 61s 9ms/step - loss: 0.4259 - acc: 0.8614 - val_lo
Epoch 65/80
7000/7000 [==============================] - 63s 9ms/step - loss: 0.4447 - acc: 0.8583 - val_lo
Epoch 66/80
7000/7000 [==============================] - 62s 9ms/step - loss: 0.4347 - acc: 0.8541 - val_lo
Epoch 67/80
7000/7000 [==============================] - 62s 9ms/step - loss: 0.4380 - acc: 0.8597 - val_lo
Epoch 68/80
7000/7000 [==============================] - 63s 9ms/step - loss: 0.4893 - acc: 0.8496 - val_lo
Epoch 69/80
7000/7000 [==============================] - 62s 9ms/step - loss: 0.5084 - acc: 0.8354 - val_lo
Epoch 70/80
7000/7000 [==============================] - 62s 9ms/step - loss: 0.4822 - acc: 0.8431 - val_lo
Epoch 71/80
7000/7000 [==============================] - 64s 9ms/step - loss: 0.4742 - acc: 0.8450 - val_lo
Epoch 72/80
7000/7000 [==============================] - 70s 10ms/step - loss: 0.4376 - acc: 0.8577 - val_l
Epoch 73/80
7000/7000 [==============================] - 72s 10ms/step - loss: 0.4144 - acc: 0.8689 - val_l
Epoch 74/80
7000/7000 [==============================] - 73s 10ms/step - loss: 0.5492 - acc: 0.8267 - val_l
Epoch 75/80
7000/7000 [==============================] - 70s 10ms/step - loss: 0.5343 - acc: 0.8346 - val_l
Epoch 76/80
7000/7000 [==============================] - 65s 9ms/step - loss: 0.4617 - acc: 0.8466 - val_lo
Epoch 77/80
7000/7000 [==============================] - 62s 9ms/step - loss: 0.4304 - acc: 0.8623 - val_lo
Epoch 78/80
7000/7000 [==============================] - 63s 9ms/step - loss: 0.4022 - acc: 0.8686 - val_lo
Epoch 79/80
7000/7000 [==============================] - 64s 9ms/step - loss: 0.4233 - acc: 0.8604 - val_lo
Epoch 80/80
7000/7000 [==============================] - 65s 9ms/step - loss: 0.3835 - acc: 0.8736 - val_lo
```


Model accuracy
Train
Test
Accuracy
Epoch
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
0
10
20
30
40
50
60
70
80


Model loss
Train
Test
Loss
Epoch
0.5
1.0
1.5
2.0
2.5
3.0
0
10
20
30
40
50
60
70
80

```
Test loss: 1.2244046683995575
Test accuracy: 0.5952890787185897
```

```
In [5]: best_model = load_model('best_model.hdf5')
```

```
In [26]: !rm data/myAudioFiles/.DS_Store
```

```
rm: data/myAudioFiles/.DS_Store: No such file or directory
```

```
In [25]: path = 'data/myAudioFiles/'

         for audio_file in sorted (os.listdir(path)):
             y, sr = librosa.load(path + audio_file, duration=2.97)
             ps = librosa.feature.melspectrogram(y=y, sr=sr)

             k = np.array(ps) #seven
             y = k.reshape(1, 128, 128, 1)
             # predict
             print('File \t{0} \tpredicted as class {1} \t{2}'.format(audio_file, best_model.pr
                   best_model.predict_classes(y)[0] == int(audio_file[0]))) # predict class

         #     plt.rcParams['figure.figsize'] = [3, 2]
         #     plt.plot(np.arange(10), best_model.predict(y).ravel())  # prediction likelihood
         #     plt.show()
```

```
File         0.air_condition.m4a         predicted as class [0]         True
File         1.car_horn_short.m4a         predicted as class [1]         True
File         2.children_playing.m4a         predicted as class [2]         True
File         4.drilling_machine.m4a         predicted as class [4]         True
File         6.gunshot_firing.m4a         predicted as class [8]         False
File         7.jackhammer_tool.m4a         predicted as class [7]         True
File         8.siren_alarm.m4a         predicted as class [8]         True
File         9.street_music.m4a         predicted as class [9]         True
```

```
In [21]: plt.rcParams['figure.figsize'] = [15, 10]  # Make output graphs larger in Jupyter Not
```

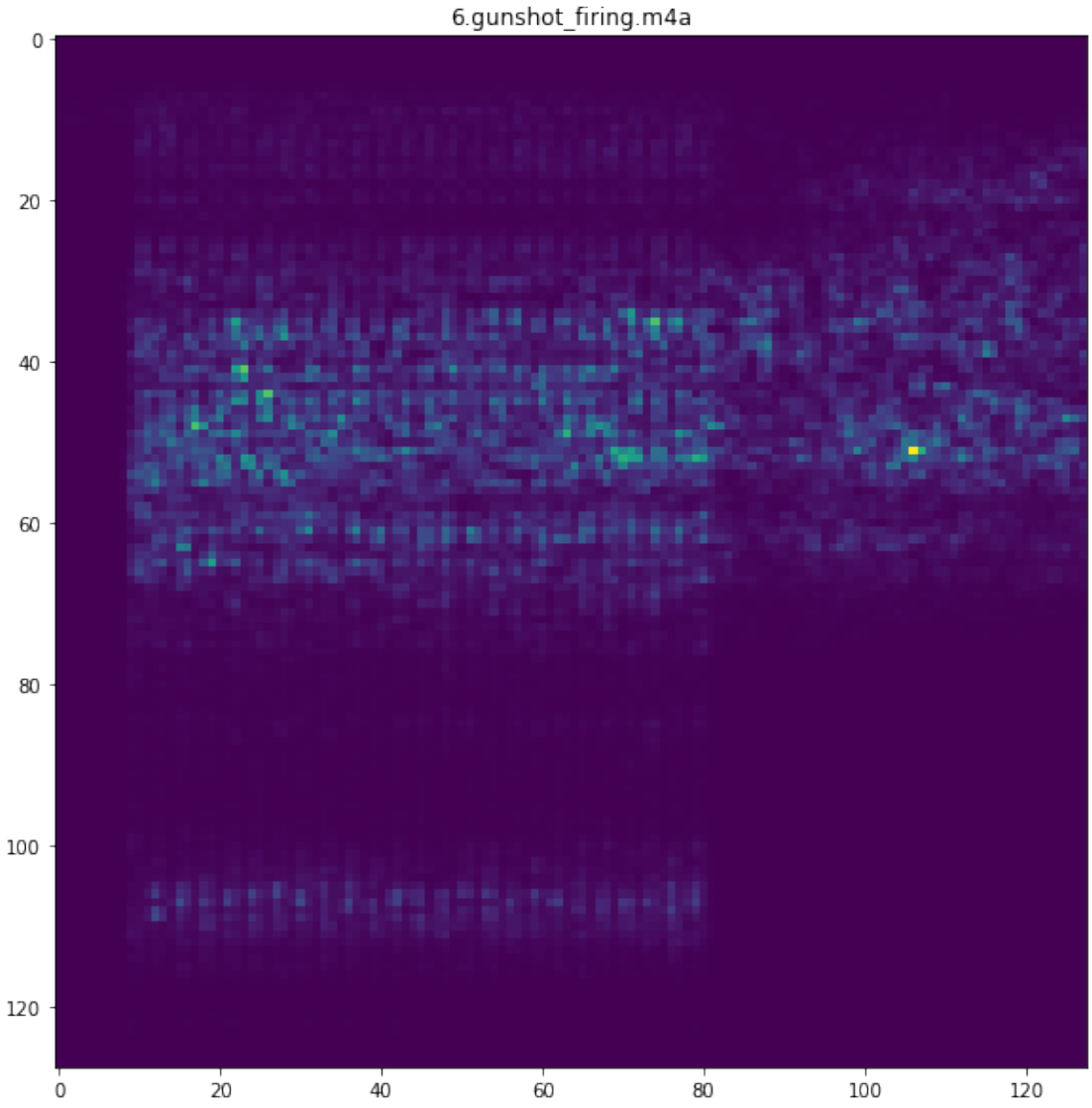
```
In [22]: !rm data/gun_firing/.DS_Store
```

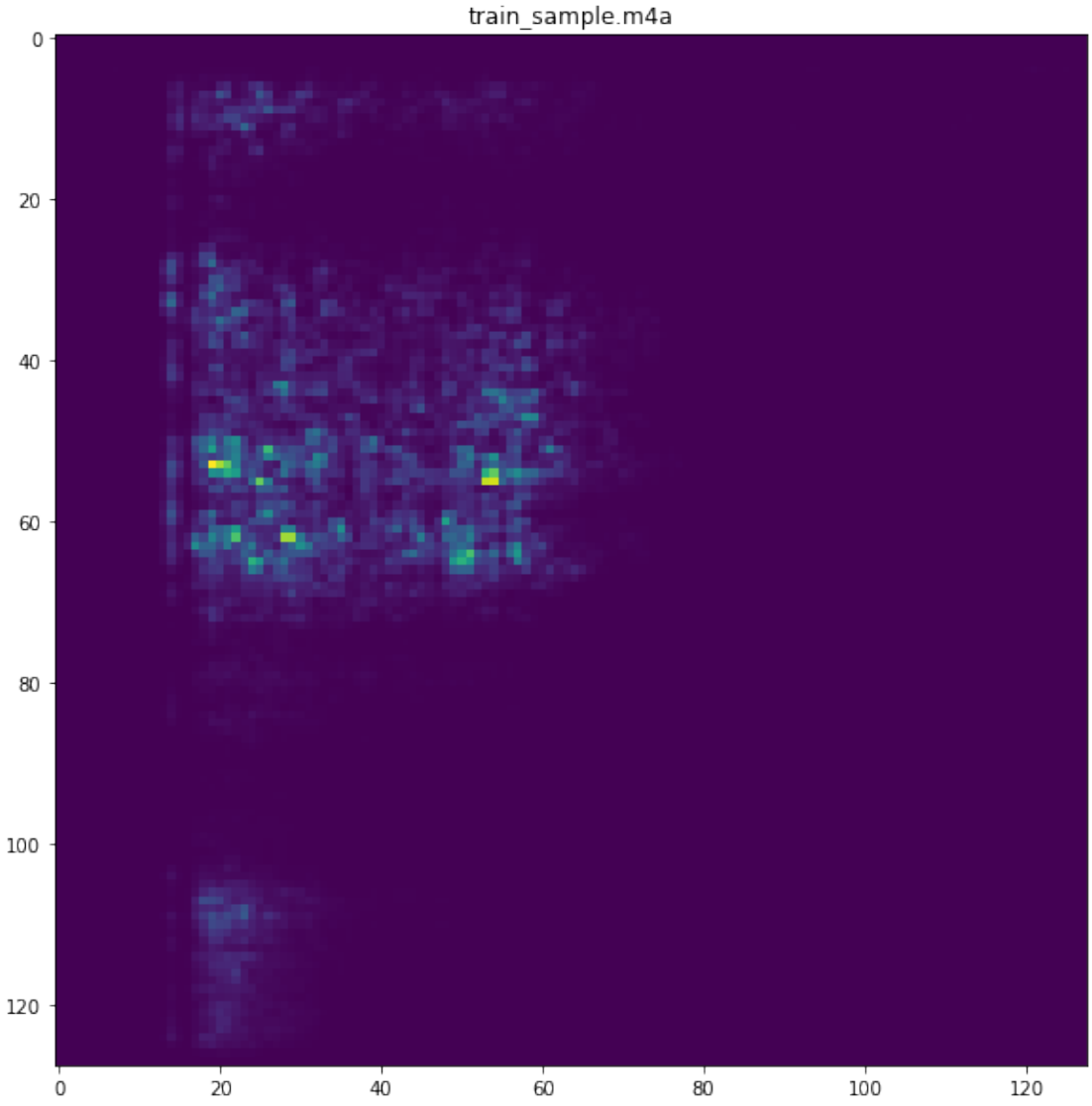
```
rm: data/gun_firing/.DS_Store: No such file or directory
```

```
In [24]: path = 'data/gun_firing/'

         for audio_file in sorted (os.listdir(path)):
             print(path + audio_file)
```

```
            y, sr = librosa.load(path + audio_file, duration=2.97)
            ps = librosa.feature.melspectrogram(y=y, sr=sr)
            k = np.array(ps)

            plt.figure(audio_file)
            plt.title(audio_file)
            plt.imshow(k)
            plt.plot()

            librosa.display.specshow(ps, y_axis='mel', x_axis='time')
```

```
data/gun_firing/6.gunshot_firing.m4a
data/gun_firing/train_sample.m4a
```




train_sample.m4a
0
20
40
60
80
100
120
0
20
40
60
80
100
120