openmic-MEL-DL-LONG

March 2, 2020

[1]:
```python
import librosa as lb
import librosa.display
import scipy
import json
import numpy as np
import sklearn
from sklearn.metrics import classification_report
from sklearn.model_selection import train_test_split
import os
import keras
from keras.utils import np_utils
from keras import layers
from keras import models
from keras.models import Sequential
from keras.layers import Dense, Conv2D, MaxPool2D , Flatten, Dropout
from keras.preprocessing.image import ImageDataGenerator
from model_builder import build_example
from plotter import plot_history
import matplotlib.pyplot as plt
```

```
Using TensorFlow backend.
```

[8]:
```python
# CONSTANTS

DATA_DIR = "openmic-2018/"
CATEGORY_COUNT = 8
LEARNING_RATE = 0.00001
THRESHOLD = 0.5
```

[4]:
```python
# LOAD DATA

OPENMIC = np.load(os.path.join(DATA_DIR, 'openmic-mel.npz'), allow_pickle=True)
print('OpenMIC keys: ' + str(list(OPENMIC.keys())))
X, Y_true, Y_mask, sample_key = OPENMIC['MEL'], OPENMIC['Y_true'],␣
↪OPENMIC['Y_mask'], OPENMIC['sample_key']
print('X has shape: ' + str(X.shape))
print('Y_true has shape: ' + str(Y_true.shape))
```

```
print('Y_mask has shape: ' + str(Y_mask.shape))
print('sample_key has shape: ' + str(sample_key.shape))
```

```
OpenMIC keys: ['MEL', 'Y_true', 'Y_mask', 'sample_key']
X has shape: (20000, 128, 430)
Y_true has shape: (20000, 20)
Y_mask has shape: (20000, 20)
sample_key has shape: (20000,)
```

[5]:
```
# LOAD LABELS

with open(os.path.join(DATA_DIR, 'class-map.json'), 'r') as f:
    INSTRUMENTS = json.load(f)
print('OpenMIC instruments: ' + str(INSTRUMENTS))
```

```
OpenMIC instruments: {'accordion': 0, 'banjo': 1, 'bass': 2, 'cello': 3,
'clarinet': 4, 'cymbals': 5, 'drums': 6, 'flute': 7, 'guitar': 8,
'mallet_percussion': 9, 'mandolin': 10, 'organ': 11, 'piano': 12, 'saxophone':
13, 'synthesizer': 14, 'trombone': 15, 'trumpet': 16, 'ukulele': 17, 'violin':
18, 'voice': 19}
```

[6]:
```
# SPLIT DATA (TRAIN - TEST - VAL)

# CHANGE X TO MEL
split_train, split_test, X_train, X_test, Y_true_train, Y_true_test,␣
 ↪Y_mask_train, Y_mask_test = train_test_split(sample_key, X, Y_true, Y_mask)
split_val, split_test, X_val, X_test, Y_true_val, Y_true_test, Y_mask_val,␣
 ↪Y_mask_test = train_test_split(split_test, X_test, Y_true_test, Y_mask_test,␣
 ↪test_size=0.5)
train_set = np.asarray(set(split_train))
test_set = np.asarray(set(split_test))
print('# Train: {}, # Val: {}, # Test: {}'.format(len(split_train),␣
 ↪len(split_test), len(split_val)))
```

```
# Train: 15000, # Val: 2500, # Test: 2500
```

[34]:
```
# DUPLICATE OF THE MODEL PREPROCESS

print(X_train.shape)
print(X_test.shape)

for instrument in INSTRUMENTS:

    # Map the instrument name to its column number
    inst_num = INSTRUMENTS[instrument]
```

```
    print(instrument)

    # TRAIN
    train_inst = Y_mask_train[:, inst_num]
    X_train_inst = X_train[train_inst]
    X_train_inst = X_train_inst.astype('float16')
    shape = X_train_inst.shape
    X_train_inst = X_train_inst.reshape(shape[0],1, shape[1], shape[2])
    Y_true_train_inst = Y_true_train[train_inst, inst_num] >= THRESHOLD
    i = 0
    for val in Y_true_train_inst:
        i += val

    print('TRAIN: ' + str(i) + ' true of ' + str(len(Y_true_train_inst)) + ' (' + str(round(i / len(Y_true_train_inst ) * 100,2)) + ' %)' )


    # TEST
    test_inst = Y_mask_test[:, inst_num]
    X_test_inst = X_test[test_inst]
    X_test_inst = X_test_inst.astype('float16')
    shape = X_test_inst.shape
    X_test_inst = X_test_inst.reshape(shape[0],1, shape[1], shape[2])
    Y_true_test_inst = Y_true_test[test_inst, inst_num] >= THRESHOLD

    i = 0
    for val in Y_true_test_inst:
        i += val

    print('TEST: ' + str(i) + ' true of ' + str(len(Y_true_test_inst)) + ' (' + str(round(i / len(Y_true_test_inst ) * 100,2)) + ' %)' )


    # VALIDATION
    val_inst = Y_mask_val[:, inst_num]
    X_val_inst = X_val[val_inst]
    X_val_inst = X_val_inst.astype('float16')
    shape = X_val_inst.shape
    X_val_inst = X_val_inst.reshape(shape[0],1, shape[1], shape[2])
    Y_true_val_inst = Y_true_val[val_inst, inst_num] >= THRESHOLD


    i = 0
    for val in Y_true_val_inst:
        i += val
    print('VALIDATION: ' + str(i) + ' true of ' + str(len(Y_true_val_inst)) + ' (' + str(round(i / len(Y_true_val_inst ) * 100,2)) + ' %)' )
```

```
(15000, 128, 430)
(2500, 128, 430)
accordion
TRAIN: 367 true of 1540 (23.83 %)
TEST: 64 true of 279 (22.94 %)
VALIDATION: 58 true of 252 (23.02 %)
banjo
TRAIN: 532 true of 1620 (32.84 %)
TEST: 99 true of 307 (32.25 %)
VALIDATION: 101 true of 291 (34.71 %)
bass
TRAIN: 410 true of 1401 (29.26 %)
TEST: 65 true of 234 (27.78 %)
VALIDATION: 74 true of 253 (29.25 %)
cello
TRAIN: 643 true of 1490 (43.15 %)
TEST: 97 true of 243 (39.92 %)
VALIDATION: 84 true of 216 (38.89 %)
clarinet
TRAIN: 411 true of 1810 (22.71 %)
TEST: 59 true of 293 (20.14 %)
VALIDATION: 63 true of 282 (22.34 %)
cymbals
TRAIN: 816 true of 1280 (63.75 %)
TEST: 144 true of 225 (64.0 %)
VALIDATION: 151 true of 230 (65.65 %)
drums
TRAIN: 827 true of 1313 (62.99 %)
TEST: 144 true of 226 (63.72 %)
VALIDATION: 135 true of 208 (64.9 %)
flute
TRAIN: 484 true of 1562 (30.99 %)
TEST: 82 true of 277 (29.6 %)
VALIDATION: 81 true of 245 (33.06 %)
guitar
TRAIN: 859 true of 1232 (69.72 %)
TEST: 147 true of 215 (68.37 %)
VALIDATION: 132 true of 203 (65.02 %)
mallet_percussion
TRAIN: 561 true of 1393 (40.27 %)
TEST: 67 true of 191 (35.08 %)
VALIDATION: 105 true of 218 (48.17 %)
mandolin
TRAIN: 619 true of 1799 (34.41 %)
TEST: 107 true of 314 (34.08 %)
VALIDATION: 119 true of 351 (33.9 %)
organ
TRAIN: 454 true of 1427 (31.81 %)
```

```
TEST: 65 true of 224 (29.02 %)
VALIDATION: 84 true of 239 (35.15 %)
piano
TRAIN: 871 true of 1284 (67.83 %)
TEST: 164 true of 237 (69.2 %)
VALIDATION: 135 true of 199 (67.84 %)
saxophone
TRAIN: 841 true of 1769 (47.54 %)
TEST: 141 true of 286 (49.3 %)
VALIDATION: 153 true of 310 (49.35 %)
synthesizer
TRAIN: 798 true of 1178 (67.74 %)
TEST: 146 true of 225 (64.89 %)
VALIDATION: 147 true of 199 (73.87 %)
trombone
TRAIN: 653 true of 2058 (31.73 %)
TEST: 110 true of 362 (30.39 %)
VALIDATION: 100 true of 340 (29.41 %)
trumpet
TRAIN: 861 true of 2179 (39.51 %)
TEST: 145 true of 385 (37.66 %)
VALIDATION: 140 true of 352 (39.77 %)
ukulele
TRAIN: 542 true of 1805 (30.03 %)
TEST: 90 true of 298 (30.2 %)
VALIDATION: 106 true of 322 (32.92 %)
violin
TRAIN: 884 true of 1529 (57.82 %)
TEST: 138 true of 244 (56.56 %)
VALIDATION: 151 true of 260 (58.08 %)
voice
TRAIN: 752 true of 1174 (64.05 %)
TEST: 126 true of 197 (63.96 %)
VALIDATION: 110 true of 193 (56.99 %)
```

[15]:
```
# VALAMI FANCY ADATKIÍRÁS
len(Y_true_val_inst)
```

[15]: 193

[15]:
```
from keras.optimizers import SGD
# This dictionary will include the classifiers for each model
mymodels = dict()
# We'll iterate over all istrument classes, and fit a model for each one
# After training, we'll print a classification report for each instrument
for instrument in INSTRUMENTS:

    # Map the instrument name to its column number
```

```
    inst_num = INSTRUMENTS[instrument]

    # Step 1: sub-sample the data
    # First, we need to select down to the data for which we have annotations
    # This is what the mask arrays are for
    # Here, we're using the Y_mask_train array to slice out only the training␣
↪examples
    # for which we have annotations for the given class
    # Again, we slice the labels to the annotated examples
    # We thresold the label likelihoods at 0.5 to get binary labels

    # TRAIN
    train_inst = Y_mask_train[:, inst_num]
    X_train_inst = X_train[train_inst]
    X_train_inst = X_train_inst.astype('float16')
    shape = X_train_inst.shape
    X_train_inst = X_train_inst.reshape(shape[0],1, shape[1], shape[2])
    Y_true_train_inst = Y_true_train[train_inst, inst_num] >= THRESHOLD

    # TEST
    test_inst = Y_mask_test[:, inst_num]
    X_test_inst = X_test[test_inst]
    X_test_inst = X_test_inst.astype('float16')
    shape = X_test_inst.shape
    X_test_inst = X_test_inst.reshape(shape[0],1, shape[1], shape[2])
    Y_true_test_inst = Y_true_test[test_inst, inst_num] >= THRESHOLD

    # VALIDATION
    val_inst = Y_mask_val[:, inst_num]
    X_val_inst = X_val[val_inst]
    X_val_inst = X_val_inst.astype('float16')
    shape = X_val_inst.shape
    X_val_inst = X_val_inst.reshape(shape[0],1, shape[1], shape[2])
    Y_true_val_inst = Y_true_val[val_inst, inst_num] >= THRESHOLD


    # Step 3.
    # Initialize a new classifier
    model = models.Sequential()
    model.
↪add(Conv2D(input_shape=(1,128,430),data_format="channels_first",filters=32,kernel_size=(3,3
↪activation="relu"))
    model.add(Conv2D(filters=32,kernel_size=(3,3),padding="same",␣
↪activation="relu"))
    model.add(MaxPool2D(pool_size=(3,3),strides=(2,2)))
    model.add(Dropout(0.25))
```

```
    model.add(Conv2D(filters=128, kernel_size=(3,3), padding="same",␣
↪activation="relu"))
    model.add(Conv2D(filters=128, kernel_size=(3,3), padding="same",␣
↪activation="relu"))
    model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
    model.add(Dropout(0.25))
    model.add(Conv2D(filters=256, kernel_size=(3,3), padding="same",␣
↪activation="relu"))
    model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
    model.add(layers.Flatten())
    model.add(layers.Dense(units=512, activation='relu'))
    model.add(layers.Dense(units=256, activation='relu'))
    model.add(layers.Dense(units=1, activation='sigmoid'))

    model.compile(loss='binary_crossentropy', optimizer=SGD(lr=0.00001),␣
↪metrics = ['accuracy'])

    # model.summary()
    # Step 4.
    history = model.fit(X_train_inst,Y_true_train_inst , epochs=50,␣
↪batch_size=32, validation_data=(X_val_inst,Y_true_val_inst))

    plot_history()

    loss, acc = model.evaluate(X_test_inst, Y_true_test_inst)
    print('Test loss: {}'.format(loss))
    print('Test accuracy: {:.2%}'.format(acc))
    # Step 5.
    # Finally, we'll evaluate the model on both train and test
    Y_pred_train = model.predict(X_train_inst)
    Y_pred_test = model.predict(X_test_inst)
    Y_pred_train_bool = Y_pred_train > THRESHOLD #THRESHOLD (should be lower␣
↪than 0.5)
    Y_pred_test_bool = Y_pred_test > THRESHOLD #THRESHOLD (should be lower than␣
↪0.5)
    print('-' * 52)
    print(instrument)
    print('\tTRAIN')
    print(classification_report(Y_true_train_inst, Y_pred_train_bool))
    print('\tTEST')
    print(classification_report(Y_true_test_inst, Y_pred_test_bool))

    # Store the classifier in our dictionary
mymodels[instrument] = model
```

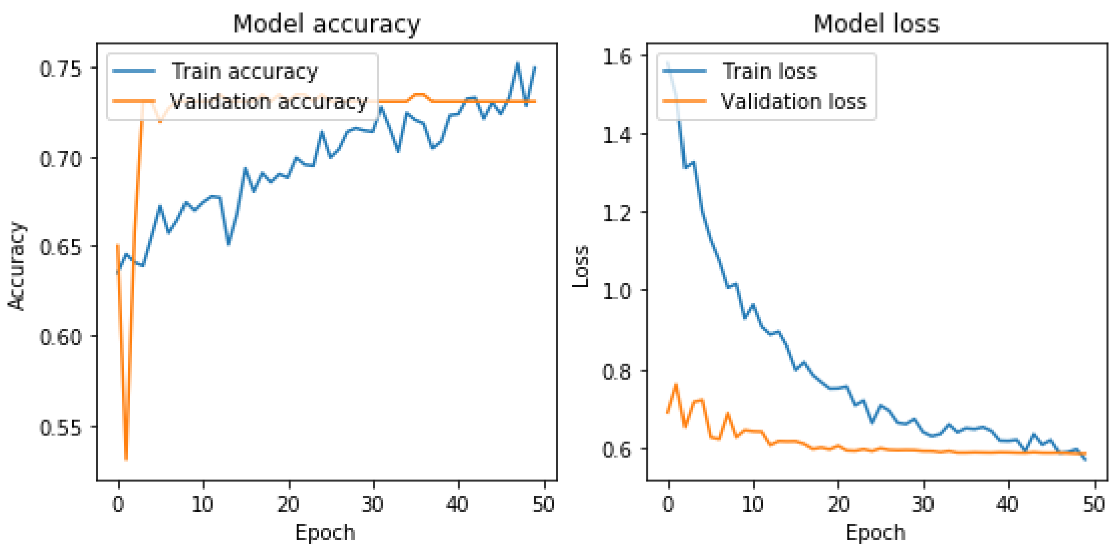
```
Train on 1520 samples, validate on 260 samples
Epoch 1/50
```

```
1520/1520 [==============================] - 353s 232ms/step - loss: 1.5777 -
acc: 0.6349 - val_loss: 0.6895 - val_acc: 0.6500
Epoch 2/50
1520/1520 [==============================] - 359s 236ms/step - loss: 1.4950 -
acc: 0.6454 - val_loss: 0.7613 - val_acc: 0.5308
Epoch 3/50
1520/1520 [==============================] - 391s 257ms/step - loss: 1.3114 -
acc: 0.6408 - val_loss: 0.6520 - val_acc: 0.6577
Epoch 4/50
1520/1520 [==============================] - 378s 249ms/step - loss: 1.3263 -
acc: 0.6388 - val_loss: 0.7163 - val_acc: 0.7308
Epoch 5/50
1520/1520 [==============================] - 383s 252ms/step - loss: 1.2001 -
acc: 0.6553 - val_loss: 0.7212 - val_acc: 0.7308
Epoch 6/50
1520/1520 [==============================] - 364s 239ms/step - loss: 1.1281 -
acc: 0.6724 - val_loss: 0.6261 - val_acc: 0.7192
Epoch 7/50
1520/1520 [==============================] - 368s 242ms/step - loss: 1.0747 -
acc: 0.6572 - val_loss: 0.6214 - val_acc: 0.7269
Epoch 8/50
1520/1520 [==============================] - 362s 238ms/step - loss: 1.0059 -
acc: 0.6645 - val_loss: 0.6872 - val_acc: 0.7308
Epoch 9/50
1520/1520 [==============================] - 353s 232ms/step - loss: 1.0157 -
acc: 0.6743 - val_loss: 0.6262 - val_acc: 0.7308
Epoch 10/50
1520/1520 [==============================] - 354s 233ms/step - loss: 0.9272 -
acc: 0.6697 - val_loss: 0.6442 - val_acc: 0.7308
Epoch 11/50
1520/1520 [==============================] - 384s 253ms/step - loss: 0.9629 -
acc: 0.6743 - val_loss: 0.6411 - val_acc: 0.7308
Epoch 12/50
1520/1520 [==============================] - 360s 237ms/step - loss: 0.9069 -
acc: 0.6776 - val_loss: 0.6403 - val_acc: 0.7308
Epoch 13/50
1520/1520 [==============================] - 356s 234ms/step - loss: 0.8864 -
acc: 0.6770 - val_loss: 0.6060 - val_acc: 0.7346
Epoch 14/50
1520/1520 [==============================] - 358s 235ms/step - loss: 0.8938 -
acc: 0.6507 - val_loss: 0.6160 - val_acc: 0.7308
Epoch 15/50
1520/1520 [==============================] - 345s 227ms/step - loss: 0.8556 -
acc: 0.6678 - val_loss: 0.6153 - val_acc: 0.7308
Epoch 16/50
1520/1520 [==============================] - 384s 253ms/step - loss: 0.7978 -
acc: 0.6934 - val_loss: 0.6156 - val_acc: 0.7308
Epoch 17/50
```

```
1520/1520 [==============================] - 379s 250ms/step - loss: 0.8180 -
acc: 0.6803 - val_loss: 0.6092 - val_acc: 0.7308
Epoch 18/50
1520/1520 [==============================] - 352s 231ms/step - loss: 0.7849 -
acc: 0.6908 - val_loss: 0.5962 - val_acc: 0.7346
Epoch 19/50
1520/1520 [==============================] - 352s 231ms/step - loss: 0.7669 -
acc: 0.6855 - val_loss: 0.6000 - val_acc: 0.7308
Epoch 20/50
1520/1520 [==============================] - 350s 230ms/step - loss: 0.7502 -
acc: 0.6901 - val_loss: 0.5954 - val_acc: 0.7346
Epoch 21/50
1520/1520 [==============================] - 353s 233ms/step - loss: 0.7509 -
acc: 0.6882 - val_loss: 0.6044 - val_acc: 0.7308
Epoch 22/50
1520/1520 [==============================] - 361s 238ms/step - loss: 0.7557 -
acc: 0.6993 - val_loss: 0.5926 - val_acc: 0.7346
Epoch 23/50
1520/1520 [==============================] - 340s 223ms/step - loss: 0.7076 -
acc: 0.6954 - val_loss: 0.5916 - val_acc: 0.7346
Epoch 24/50
1520/1520 [==============================] - 342s 225ms/step - loss: 0.7200 -
acc: 0.6947 - val_loss: 0.5957 - val_acc: 0.7308
Epoch 25/50
1520/1520 [==============================] - 338s 223ms/step - loss: 0.6623 -
acc: 0.7138 - val_loss: 0.5909 - val_acc: 0.7346
Epoch 26/50
1520/1520 [==============================] - 341s 224ms/step - loss: 0.7073 -
acc: 0.6993 - val_loss: 0.5981 - val_acc: 0.7308
Epoch 27/50
1520/1520 [==============================] - 340s 224ms/step - loss: 0.6942 -
acc: 0.7039 - val_loss: 0.5940 - val_acc: 0.7308
Epoch 28/50
1520/1520 [==============================] - 341s 224ms/step - loss: 0.6631 -
acc: 0.7138 - val_loss: 0.5932 - val_acc: 0.7308
Epoch 29/50
1520/1520 [==============================] - 343s 226ms/step - loss: 0.6598 -
acc: 0.7158 - val_loss: 0.5936 - val_acc: 0.7308
Epoch 30/50
1520/1520 [==============================] - 339s 223ms/step - loss: 0.6727 -
acc: 0.7145 - val_loss: 0.5932 - val_acc: 0.7308
Epoch 31/50
1520/1520 [==============================] - 341s 224ms/step - loss: 0.6396 -
acc: 0.7138 - val_loss: 0.5911 - val_acc: 0.7308
Epoch 32/50
1520/1520 [==============================] - 341s 224ms/step - loss: 0.6290 -
acc: 0.7276 - val_loss: 0.5909 - val_acc: 0.7308
Epoch 33/50
```

```
1520/1520 [==============================] - 346s 228ms/step - loss: 0.6344 -
acc: 0.7158 - val_loss: 0.5882 - val_acc: 0.7308
Epoch 34/50
1520/1520 [==============================] - 342s 225ms/step - loss: 0.6586 -
acc: 0.7026 - val_loss: 0.5911 - val_acc: 0.7308
Epoch 35/50
1520/1520 [==============================] - 340s 224ms/step - loss: 0.6388 -
acc: 0.7243 - val_loss: 0.5868 - val_acc: 0.7308
Epoch 36/50
1520/1520 [==============================] - 339s 223ms/step - loss: 0.6492 -
acc: 0.7204 - val_loss: 0.5868 - val_acc: 0.7346
Epoch 37/50
1520/1520 [==============================] - 341s 225ms/step - loss: 0.6468 -
acc: 0.7184 - val_loss: 0.5876 - val_acc: 0.7346
Epoch 38/50
1520/1520 [==============================] - 343s 226ms/step - loss: 0.6518 -
acc: 0.7046 - val_loss: 0.5871 - val_acc: 0.7308
Epoch 39/50
1520/1520 [==============================] - 341s 224ms/step - loss: 0.6423 -
acc: 0.7086 - val_loss: 0.5868 - val_acc: 0.7308
Epoch 40/50
1520/1520 [==============================] - 342s 225ms/step - loss: 0.6173 -
acc: 0.7230 - val_loss: 0.5877 - val_acc: 0.7308
Epoch 41/50
1520/1520 [==============================] - 340s 224ms/step - loss: 0.6165 -
acc: 0.7237 - val_loss: 0.5874 - val_acc: 0.7308
Epoch 42/50
1520/1520 [==============================] - 339s 223ms/step - loss: 0.6193 -
acc: 0.7322 - val_loss: 0.5866 - val_acc: 0.7308
Epoch 43/50
1520/1520 [==============================] - 342s 225ms/step - loss: 0.5906 -
acc: 0.7329 - val_loss: 0.5861 - val_acc: 0.7308
Epoch 44/50
1520/1520 [==============================] - 339s 223ms/step - loss: 0.6342 -
acc: 0.7211 - val_loss: 0.5879 - val_acc: 0.7308
Epoch 45/50
1520/1520 [==============================] - 345s 227ms/step - loss: 0.6071 -
acc: 0.7303 - val_loss: 0.5864 - val_acc: 0.7308
Epoch 46/50
1520/1520 [==============================] - 340s 224ms/step - loss: 0.6187 -
acc: 0.7237 - val_loss: 0.5862 - val_acc: 0.7308
Epoch 47/50
1520/1520 [==============================] - 342s 225ms/step - loss: 0.5860 -
acc: 0.7329 - val_loss: 0.5865 - val_acc: 0.7308
Epoch 48/50
1520/1520 [==============================] - 339s 223ms/step - loss: 0.5886 -
acc: 0.7520 - val_loss: 0.5853 - val_acc: 0.7308
Epoch 49/50
```

```
1520/1520 [==============================] - 340s 224ms/step - loss: 0.5957 -
acc: 0.7283 - val_loss: 0.5840 - val_acc: 0.7308
Epoch 50/50
1520/1520 [==============================] - 342s 225ms/step - loss: 0.5693 -
acc: 0.7493 - val_loss: 0.5845 - val_acc: 0.7308
```



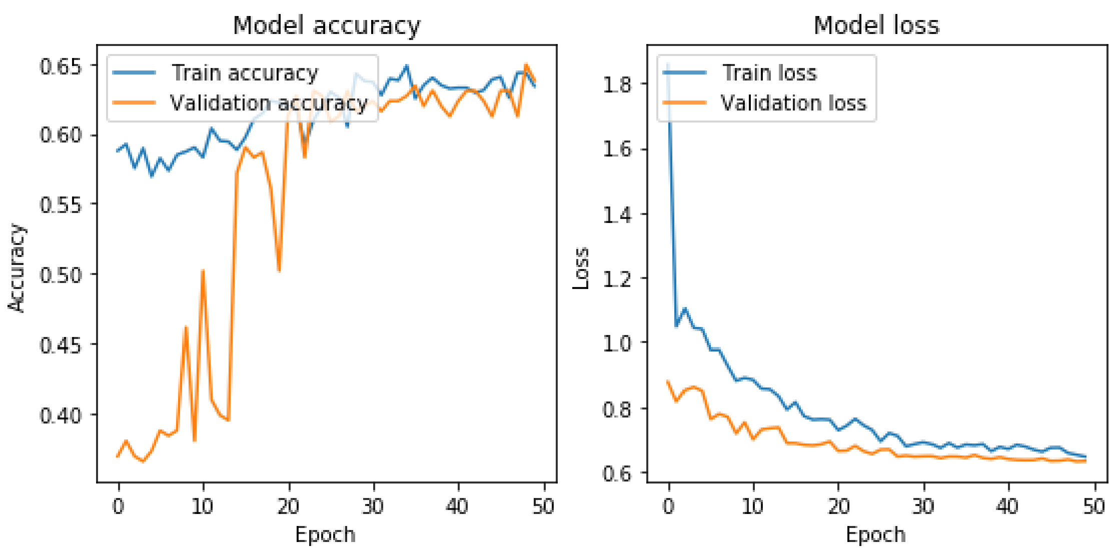

```
291/291 [==============================] - 25s 85ms/step
Test loss: 0.5564455607093077
Test accuracy: 76.98%
----------------------------------------------------
accordion
        TRAIN
              precision    recall  f1-score   support

       False       0.77      1.00      0.87      1168
        True       0.00      0.00      0.00       352

    accuracy                           0.77      1520
   macro avg       0.38      0.50      0.43      1520
weighted avg       0.59      0.77      0.67      1520

        TEST
              precision    recall  f1-score   support

       False       0.77      1.00      0.87       224
        True       0.00      0.00      0.00        67

    accuracy                           0.77       291
   macro avg       0.38      0.50      0.43       291
```

```
weighted avg       0.59      0.77      0.67       291


C:\Users\hjani\Documents\Conda\lib\site-
packages\sklearn\metrics\classification.py:1437: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples.
  'precision', 'predicted', average, warn_for)

Train on 1679 samples, validate on 271 samples
Epoch 1/50
1679/1679 [==============================] - 381s 227ms/step - loss: 1.8556 -
acc: 0.5878 - val_loss: 0.8777 - val_acc: 0.3690
Epoch 2/50
1679/1679 [==============================] - 371s 221ms/step - loss: 1.0476 -
acc: 0.5926 - val_loss: 0.8169 - val_acc: 0.3801
Epoch 3/50
1679/1679 [==============================] - 371s 221ms/step - loss: 1.1032 -
acc: 0.5753 - val_loss: 0.8527 - val_acc: 0.3690
Epoch 4/50
1679/1679 [==============================] - 370s 220ms/step - loss: 1.0434 -
acc: 0.5896 - val_loss: 0.8618 - val_acc: 0.3653
Epoch 5/50
1679/1679 [==============================] - 373s 222ms/step - loss: 1.0396 -
acc: 0.5694 - val_loss: 0.8500 - val_acc: 0.3727
Epoch 6/50
1679/1679 [==============================] - 370s 221ms/step - loss: 0.9761 -
acc: 0.5825 - val_loss: 0.7623 - val_acc: 0.3875
Epoch 7/50
1679/1679 [==============================] - 370s 221ms/step - loss: 0.9765 -
acc: 0.5736 - val_loss: 0.7785 - val_acc: 0.3838
Epoch 8/50
1679/1679 [==============================] - 370s 220ms/step - loss: 0.9276 -
acc: 0.5849 - val_loss: 0.7694 - val_acc: 0.3875
Epoch 9/50
1679/1679 [==============================] - 370s 220ms/step - loss: 0.8809 -
acc: 0.5873 - val_loss: 0.7181 - val_acc: 0.4613
Epoch 10/50
1679/1679 [==============================] - 371s 221ms/step - loss: 0.8901 -
acc: 0.5902 - val_loss: 0.7525 - val_acc: 0.3801
Epoch 11/50
1679/1679 [==============================] - 370s 220ms/step - loss: 0.8840 -
acc: 0.5831 - val_loss: 0.6997 - val_acc: 0.5018
Epoch 12/50
1679/1679 [==============================] - 371s 221ms/step - loss: 0.8567 -
acc: 0.6039 - val_loss: 0.7303 - val_acc: 0.4096
Epoch 13/50
1679/1679 [==============================] - 369s 220ms/step - loss: 0.8548 -
```

```
acc: 0.5950 - val_loss: 0.7342 - val_acc: 0.3985
Epoch 14/50
1679/1679 [==============================] - 376s 224ms/step - loss: 0.8334 -
acc: 0.5944 - val_loss: 0.7365 - val_acc: 0.3948
Epoch 15/50
1679/1679 [==============================] - 369s 220ms/step - loss: 0.7919 -
acc: 0.5884 - val_loss: 0.6882 - val_acc: 0.5720
Epoch 16/50
1679/1679 [==============================] - 370s 220ms/step - loss: 0.8148 -
acc: 0.5974 - val_loss: 0.6878 - val_acc: 0.5904
Epoch 17/50
1679/1679 [==============================] - 370s 220ms/step - loss: 0.7720 -
acc: 0.6105 - val_loss: 0.6828 - val_acc: 0.5830
Epoch 18/50
1679/1679 [==============================] - 370s 220ms/step - loss: 0.7606 -
acc: 0.6147 - val_loss: 0.6811 - val_acc: 0.5867
Epoch 19/50
1679/1679 [==============================] - 377s 224ms/step - loss: 0.7621 -
acc: 0.6236 - val_loss: 0.6845 - val_acc: 0.5609
Epoch 20/50
1679/1679 [==============================] - 372s 222ms/step - loss: 0.7607 -
acc: 0.6224 - val_loss: 0.6932 - val_acc: 0.5018
Epoch 21/50
1679/1679 [==============================] - 370s 221ms/step - loss: 0.7281 -
acc: 0.6200 - val_loss: 0.6639 - val_acc: 0.6125
Epoch 22/50
1679/1679 [==============================] - 372s 221ms/step - loss: 0.7421 -
acc: 0.6194 - val_loss: 0.6650 - val_acc: 0.6273
Epoch 23/50
1679/1679 [==============================] - 370s 220ms/step - loss: 0.7633 -
acc: 0.5926 - val_loss: 0.6797 - val_acc: 0.5830
Epoch 24/50
1679/1679 [==============================] - 372s 221ms/step - loss: 0.7430 -
acc: 0.6099 - val_loss: 0.6631 - val_acc: 0.6310
Epoch 25/50
1679/1679 [==============================] - 369s 220ms/step - loss: 0.7288 -
acc: 0.6200 - val_loss: 0.6547 - val_acc: 0.6273
Epoch 26/50
1679/1679 [==============================] - 370s 220ms/step - loss: 0.6938 -
acc: 0.6301 - val_loss: 0.6680 - val_acc: 0.6089
Epoch 27/50
1679/1679 [==============================] - 370s 221ms/step - loss: 0.7192 -
acc: 0.6260 - val_loss: 0.6688 - val_acc: 0.6125
Epoch 28/50
1679/1679 [==============================] - 410s 244ms/step - loss: 0.7103 -
acc: 0.6051 - val_loss: 0.6467 - val_acc: 0.6310
Epoch 29/50
1679/1679 [==============================] - 373s 222ms/step - loss: 0.6781 -
```

```
acc: 0.6432 - val_loss: 0.6495 - val_acc: 0.6162
Epoch 30/50
1679/1679 [==============================] - 373s 222ms/step - loss: 0.6849 -
acc: 0.6379 - val_loss: 0.6462 - val_acc: 0.6199
Epoch 31/50
1679/1679 [==============================] - 375s 223ms/step - loss: 0.6907 -
acc: 0.6373 - val_loss: 0.6475 - val_acc: 0.6236
Epoch 32/50
1679/1679 [==============================] - 375s 223ms/step - loss: 0.6850 -
acc: 0.6278 - val_loss: 0.6479 - val_acc: 0.6162
Epoch 33/50
1679/1679 [==============================] - 372s 222ms/step - loss: 0.6731 -
acc: 0.6397 - val_loss: 0.6418 - val_acc: 0.6236
Epoch 34/50
1679/1679 [==============================] - 374s 223ms/step - loss: 0.6882 -
acc: 0.6385 - val_loss: 0.6468 - val_acc: 0.6236
Epoch 35/50
1679/1679 [==============================] - 372s 222ms/step - loss: 0.6741 -
acc: 0.6486 - val_loss: 0.6465 - val_acc: 0.6273
Epoch 36/50
1679/1679 [==============================] - 373s 222ms/step - loss: 0.6839 -
acc: 0.6254 - val_loss: 0.6432 - val_acc: 0.6347
Epoch 37/50
1679/1679 [==============================] - 372s 222ms/step - loss: 0.6812 -
acc: 0.6349 - val_loss: 0.6513 - val_acc: 0.6199
Epoch 38/50
1679/1679 [==============================] - 372s 221ms/step - loss: 0.6851 -
acc: 0.6403 - val_loss: 0.6423 - val_acc: 0.6310
Epoch 39/50
1679/1679 [==============================] - 374s 223ms/step - loss: 0.6631 -
acc: 0.6349 - val_loss: 0.6394 - val_acc: 0.6199
Epoch 40/50
1679/1679 [==============================] - 375s 223ms/step - loss: 0.6764 -
acc: 0.6325 - val_loss: 0.6446 - val_acc: 0.6125
Epoch 41/50
1679/1679 [==============================] - 375s 223ms/step - loss: 0.6699 -
acc: 0.6331 - val_loss: 0.6388 - val_acc: 0.6236
Epoch 42/50
1679/1679 [==============================] - 373s 222ms/step - loss: 0.6831 -
acc: 0.6331 - val_loss: 0.6362 - val_acc: 0.6310
Epoch 43/50
1679/1679 [==============================] - 381s 227ms/step - loss: 0.6772 -
acc: 0.6295 - val_loss: 0.6354 - val_acc: 0.6310
Epoch 44/50
1679/1679 [==============================] - 371s 221ms/step - loss: 0.6679 -
acc: 0.6313 - val_loss: 0.6359 - val_acc: 0.6236
Epoch 45/50
1679/1679 [==============================] - 374s 223ms/step - loss: 0.6615 -
```

```
acc: 0.6391 - val_loss: 0.6408 - val_acc: 0.6125
Epoch 46/50
1679/1679 [==============================] - 373s 222ms/step - loss: 0.6736 -
acc: 0.6409 - val_loss: 0.6330 - val_acc: 0.6310
Epoch 47/50
1679/1679 [==============================] - 371s 221ms/step - loss: 0.6743 -
acc: 0.6260 - val_loss: 0.6338 - val_acc: 0.6310
Epoch 48/50
1679/1679 [==============================] - 373s 222ms/step - loss: 0.6574 -
acc: 0.6438 - val_loss: 0.6375 - val_acc: 0.6125
Epoch 49/50
1679/1679 [==============================] - 373s 222ms/step - loss: 0.6518 -
acc: 0.6438 - val_loss: 0.6316 - val_acc: 0.6494
Epoch 50/50
1679/1679 [==============================] - 375s 223ms/step - loss: 0.6463 -
acc: 0.6343 - val_loss: 0.6327 - val_acc: 0.6384
```



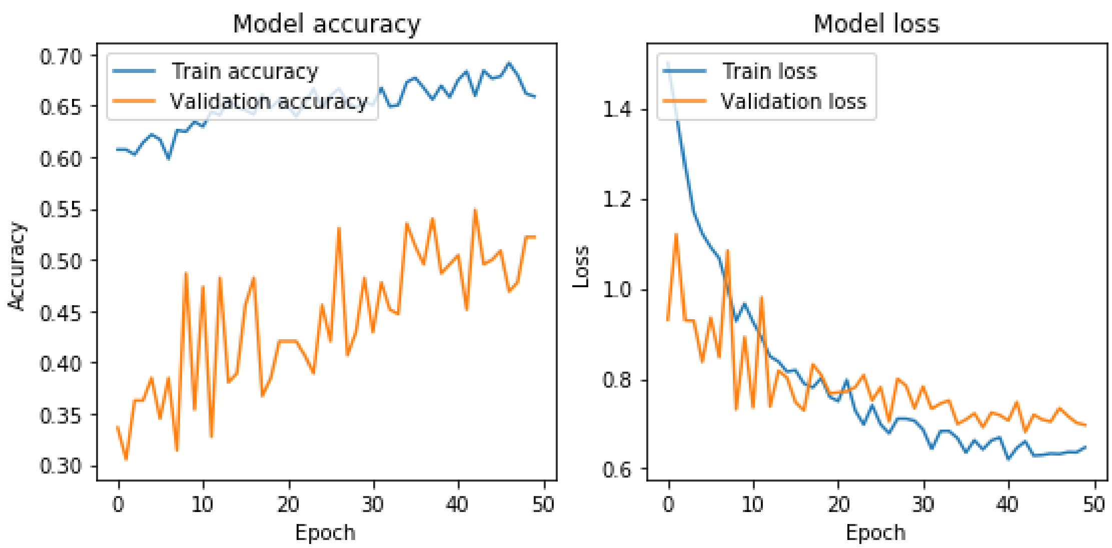

```
268/268 [==============================] - 22s 81ms/step
Test loss: 0.6290143612605422
Test accuracy: 65.67%
----------------------------------------------------
banjo
        TRAIN
              precision    recall  f1-score   support

       False       0.69      0.93      0.79      1133
        True       0.49      0.13      0.21       546

    accuracy                           0.67      1679
```

```
   macro avg       0.59      0.53      0.50      1679
weighted avg       0.63      0.67      0.60      1679

        TEST
              precision    recall  f1-score   support

       False       0.68      0.92      0.78       179
        True       0.44      0.13      0.21        89

    accuracy                           0.66       268
   macro avg       0.56      0.53      0.49       268
weighted avg       0.60      0.66      0.59       268

Train on 1425 samples, validate on 226 samples
Epoch 1/50
1425/1425 [==============================] - 317s 222ms/step - loss: 1.4996 -
acc: 0.6070 - val_loss: 0.9306 - val_acc: 0.3363
Epoch 2/50
1425/1425 [==============================] - 316s 221ms/step - loss: 1.3878 -
acc: 0.6070 - val_loss: 1.1208 - val_acc: 0.3053
Epoch 3/50
1425/1425 [==============================] - 316s 222ms/step - loss: 1.2721 -
acc: 0.6021 - val_loss: 0.9290 - val_acc: 0.3628
Epoch 4/50
1425/1425 [==============================] - 315s 221ms/step - loss: 1.1700 -
acc: 0.6140 - val_loss: 0.9278 - val_acc: 0.3628
Epoch 5/50
1425/1425 [==============================] - 315s 221ms/step - loss: 1.1223 -
acc: 0.6218 - val_loss: 0.8375 - val_acc: 0.3850
Epoch 6/50
1425/1425 [==============================] - 316s 221ms/step - loss: 1.0918 -
acc: 0.6168 - val_loss: 0.9350 - val_acc: 0.3451
Epoch 7/50
1425/1425 [==============================] - 314s 220ms/step - loss: 1.0671 -
acc: 0.5979 - val_loss: 0.8477 - val_acc: 0.3850
Epoch 8/50
1425/1425 [==============================] - 330s 232ms/step - loss: 0.9994 -
acc: 0.6260 - val_loss: 1.0843 - val_acc: 0.3142
Epoch 9/50
1425/1425 [==============================] - 316s 222ms/step - loss: 0.9271 -
acc: 0.6246 - val_loss: 0.7317 - val_acc: 0.4867
Epoch 10/50
1425/1425 [==============================] - 312s 219ms/step - loss: 0.9659 -
acc: 0.6344 - val_loss: 0.8918 - val_acc: 0.3540
Epoch 11/50
1425/1425 [==============================] - 320s 225ms/step - loss: 0.9252 -
acc: 0.6295 - val_loss: 0.7357 - val_acc: 0.4735
Epoch 12/50
```

```
1425/1425 [==============================] - 331s 232ms/step - loss: 0.8903 -
acc: 0.6442 - val_loss: 0.9799 - val_acc: 0.3274
Epoch 13/50
1425/1425 [==============================] - 318s 223ms/step - loss: 0.8496 -
acc: 0.6407 - val_loss: 0.7377 - val_acc: 0.4823
Epoch 14/50
1425/1425 [==============================] - 320s 225ms/step - loss: 0.8381 -
acc: 0.6561 - val_loss: 0.8176 - val_acc: 0.3805
Epoch 15/50
1425/1425 [==============================] - 318s 223ms/step - loss: 0.8156 -
acc: 0.6484 - val_loss: 0.8021 - val_acc: 0.3894
Epoch 16/50
1425/1425 [==============================] - 318s 223ms/step - loss: 0.8190 -
acc: 0.6456 - val_loss: 0.7475 - val_acc: 0.4558
Epoch 17/50
1425/1425 [==============================] - 320s 225ms/step - loss: 0.7894 -
acc: 0.6414 - val_loss: 0.7291 - val_acc: 0.4823
Epoch 18/50
1425/1425 [==============================] - 317s 222ms/step - loss: 0.7803 -
acc: 0.6611 - val_loss: 0.8320 - val_acc: 0.3673
Epoch 19/50
1425/1425 [==============================] - 322s 226ms/step - loss: 0.8014 -
acc: 0.6477 - val_loss: 0.8078 - val_acc: 0.3850
Epoch 20/50
1425/1425 [==============================] - 317s 222ms/step - loss: 0.7593 -
acc: 0.6554 - val_loss: 0.7671 - val_acc: 0.4204
Epoch 21/50
1425/1425 [==============================] - 316s 222ms/step - loss: 0.7499 -
acc: 0.6498 - val_loss: 0.7694 - val_acc: 0.4204
Epoch 22/50
1425/1425 [==============================] - 319s 224ms/step - loss: 0.7972 -
acc: 0.6393 - val_loss: 0.7711 - val_acc: 0.4204
Epoch 23/50
1425/1425 [==============================] - 318s 223ms/step - loss: 0.7290 -
acc: 0.6519 - val_loss: 0.7809 - val_acc: 0.4071
Epoch 24/50
1425/1425 [==============================] - 316s 222ms/step - loss: 0.6979 -
acc: 0.6667 - val_loss: 0.8079 - val_acc: 0.3894
Epoch 25/50
1425/1425 [==============================] - 324s 228ms/step - loss: 0.7408 -
acc: 0.6477 - val_loss: 0.7514 - val_acc: 0.4558
Epoch 26/50
1425/1425 [==============================] - 319s 224ms/step - loss: 0.6980 -
acc: 0.6596 - val_loss: 0.7810 - val_acc: 0.4204
Epoch 27/50
1425/1425 [==============================] - 316s 221ms/step - loss: 0.6783 -
acc: 0.6667 - val_loss: 0.7045 - val_acc: 0.5310
Epoch 28/50
```

```
1425/1425 [==============================] - 319s 224ms/step - loss: 0.7110 -
acc: 0.6533 - val_loss: 0.7997 - val_acc: 0.4071
Epoch 29/50
1425/1425 [==============================] - 316s 222ms/step - loss: 0.7109 -
acc: 0.6477 - val_loss: 0.7843 - val_acc: 0.4292
Epoch 30/50
1425/1425 [==============================] - 318s 223ms/step - loss: 0.7063 -
acc: 0.6540 - val_loss: 0.7339 - val_acc: 0.4823
Epoch 31/50
1425/1425 [==============================] - 319s 224ms/step - loss: 0.6869 -
acc: 0.6505 - val_loss: 0.7826 - val_acc: 0.4292
Epoch 32/50
1425/1425 [==============================] - 316s 222ms/step - loss: 0.6441 -
acc: 0.6674 - val_loss: 0.7332 - val_acc: 0.4779
Epoch 33/50
1425/1425 [==============================] - 318s 223ms/step - loss: 0.6831 -
acc: 0.6491 - val_loss: 0.7446 - val_acc: 0.4513
Epoch 34/50
1425/1425 [==============================] - 318s 223ms/step - loss: 0.6836 -
acc: 0.6505 - val_loss: 0.7515 - val_acc: 0.4469
Epoch 35/50
1425/1425 [==============================] - 315s 221ms/step - loss: 0.6677 -
acc: 0.6730 - val_loss: 0.6990 - val_acc: 0.5354
Epoch 36/50
1425/1425 [==============================] - 317s 223ms/step - loss: 0.6353 -
acc: 0.6772 - val_loss: 0.7091 - val_acc: 0.5133
Epoch 37/50
1425/1425 [==============================] - 318s 223ms/step - loss: 0.6630 -
acc: 0.6674 - val_loss: 0.7226 - val_acc: 0.4956
Epoch 38/50
1425/1425 [==============================] - 316s 222ms/step - loss: 0.6424 -
acc: 0.6561 - val_loss: 0.6919 - val_acc: 0.5398
Epoch 39/50
1425/1425 [==============================] - 319s 224ms/step - loss: 0.6623 -
acc: 0.6695 - val_loss: 0.7244 - val_acc: 0.4867
Epoch 40/50
1425/1425 [==============================] - 316s 222ms/step - loss: 0.6692 -
acc: 0.6582 - val_loss: 0.7187 - val_acc: 0.4956
Epoch 41/50
1425/1425 [==============================] - 315s 221ms/step - loss: 0.6202 -
acc: 0.6751 - val_loss: 0.7069 - val_acc: 0.5044
Epoch 42/50
1425/1425 [==============================] - 320s 225ms/step - loss: 0.6455 -
acc: 0.6835 - val_loss: 0.7478 - val_acc: 0.4513
Epoch 43/50
1425/1425 [==============================] - 315s 221ms/step - loss: 0.6604 -
acc: 0.6596 - val_loss: 0.6814 - val_acc: 0.5487
Epoch 44/50
```

```
1425/1425 [==============================] - 318s 223ms/step - loss: 0.6287 -
acc: 0.6842 - val_loss: 0.7197 - val_acc: 0.4956
Epoch 45/50
1425/1425 [==============================] - 321s 225ms/step - loss: 0.6301 -
acc: 0.6765 - val_loss: 0.7089 - val_acc: 0.5000
Epoch 46/50
1425/1425 [==============================] - 316s 222ms/step - loss: 0.6332 -
acc: 0.6786 - val_loss: 0.7043 - val_acc: 0.5088
Epoch 47/50
1425/1425 [==============================] - 317s 222ms/step - loss: 0.6324 -
acc: 0.6912 - val_loss: 0.7343 - val_acc: 0.4690
Epoch 48/50
1425/1425 [==============================] - 321s 225ms/step - loss: 0.6366 -
acc: 0.6800 - val_loss: 0.7165 - val_acc: 0.4779
Epoch 49/50
1425/1425 [==============================] - 316s 222ms/step - loss: 0.6360 -
acc: 0.6618 - val_loss: 0.7016 - val_acc: 0.5221
Epoch 50/50
1425/1425 [==============================] - 318s 223ms/step - loss: 0.6470 -
acc: 0.6589 - val_loss: 0.6970 - val_acc: 0.5221
```




```
237/237 [==============================] - 19s 82ms/step
Test loss: 0.7170468908322009
Test accuracy: 48.95%
----------------------------------------------------
bass
        TRAIN
              precision    recall  f1-score   support
```

```
       False       0.93      0.36      0.52      1001
        True       0.38      0.94      0.54       424

    accuracy                           0.53      1425
   macro avg       0.66      0.65      0.53      1425
weighted avg       0.77      0.53      0.52      1425

        TEST
              precision    recall  f1-score   support

       False       0.95      0.33      0.49       177
        True       0.33      0.95      0.49        60

    accuracy                           0.49       237
   macro avg       0.64      0.64      0.49       237
weighted avg       0.79      0.49      0.49       237

Train on 1436 samples, validate on 263 samples
Epoch 1/50
1436/1436 [==============================] - 322s 224ms/step - loss: 1.6675 -
acc: 0.5014 - val_loss: 1.6687 - val_acc: 0.5361
Epoch 2/50
1436/1436 [==============================] - 322s 224ms/step - loss: 1.4129 -
acc: 0.5195 - val_loss: 1.0978 - val_acc: 0.5361
Epoch 3/50
1436/1436 [==============================] - 326s 227ms/step - loss: 1.3957 -
acc: 0.5084 - val_loss: 0.9982 - val_acc: 0.5361
Epoch 4/50
1436/1436 [==============================] - 321s 224ms/step - loss: 1.3209 -
acc: 0.4937 - val_loss: 0.9357 - val_acc: 0.5361
Epoch 5/50
1436/1436 [==============================] - 325s 226ms/step - loss: 1.1832 -
acc: 0.5292 - val_loss: 0.9245 - val_acc: 0.5361
Epoch 6/50
1436/1436 [==============================] - 325s 226ms/step - loss: 1.1165 -
acc: 0.5327 - val_loss: 0.7784 - val_acc: 0.5323
Epoch 7/50
1436/1436 [==============================] - 321s 223ms/step - loss: 1.1430 -
acc: 0.5230 - val_loss: 0.9316 - val_acc: 0.5361
Epoch 8/50
1436/1436 [==============================] - 325s 226ms/step - loss: 1.0510 -
acc: 0.5334 - val_loss: 0.9168 - val_acc: 0.5361
Epoch 9/50
1436/1436 [==============================] - 330s 230ms/step - loss: 1.0130 -
acc: 0.5529 - val_loss: 0.9815 - val_acc: 0.5361
Epoch 10/50
1436/1436 [==============================] - 321s 223ms/step - loss: 0.9586 -
acc: 0.5515 - val_loss: 0.8009 - val_acc: 0.5361
```

```
Epoch 11/50
1436/1436 [==============================] - 326s 227ms/step - loss: 0.9796 -
acc: 0.5453 - val_loss: 0.8218 - val_acc: 0.5361
Epoch 12/50
1436/1436 [==============================] - 322s 224ms/step - loss: 0.9675 -
acc: 0.5299 - val_loss: 0.7960 - val_acc: 0.5361
Epoch 13/50
1436/1436 [==============================] - 322s 224ms/step - loss: 0.9211 -
acc: 0.5411 - val_loss: 0.8719 - val_acc: 0.5361
Epoch 14/50
1436/1436 [==============================] - 324s 225ms/step - loss: 0.9020 -
acc: 0.5536 - val_loss: 0.7682 - val_acc: 0.5361
Epoch 15/50
1436/1436 [==============================] - 321s 224ms/step - loss: 0.9011 -
acc: 0.5439 - val_loss: 0.8190 - val_acc: 0.5361
Epoch 16/50
1436/1436 [==============================] - 322s 225ms/step - loss: 0.8753 -
acc: 0.5557 - val_loss: 0.7785 - val_acc: 0.5361
Epoch 17/50
1436/1436 [==============================] - 324s 225ms/step - loss: 0.8426 -
acc: 0.5606 - val_loss: 0.7877 - val_acc: 0.5361
Epoch 18/50
1436/1436 [==============================] - 323s 225ms/step - loss: 0.8290 -
acc: 0.5606 - val_loss: 0.7507 - val_acc: 0.5361
Epoch 19/50
1436/1436 [==============================] - 325s 226ms/step - loss: 0.8765 -
acc: 0.5320 - val_loss: 0.7456 - val_acc: 0.5361
Epoch 20/50
1436/1436 [==============================] - 324s 226ms/step - loss: 0.8268 -
acc: 0.5564 - val_loss: 0.7399 - val_acc: 0.5361
Epoch 21/50
1436/1436 [==============================] - 321s 223ms/step - loss: 0.7900 -
acc: 0.5627 - val_loss: 0.7214 - val_acc: 0.5361
Epoch 22/50
1436/1436 [==============================] - 324s 226ms/step - loss: 0.7826 -
acc: 0.5578 - val_loss: 0.7410 - val_acc: 0.5361
Epoch 23/50
1436/1436 [==============================] - 321s 224ms/step - loss: 0.7893 -
acc: 0.5801 - val_loss: 0.7165 - val_acc: 0.5285
Epoch 24/50
1436/1436 [==============================] - 322s 224ms/step - loss: 0.7828 -
acc: 0.5669 - val_loss: 0.7348 - val_acc: 0.5361
Epoch 25/50
1436/1436 [==============================] - 324s 226ms/step - loss: 0.7637 -
acc: 0.5634 - val_loss: 0.6900 - val_acc: 0.5399
Epoch 26/50
1436/1436 [==============================] - 321s 224ms/step - loss: 0.7765 -
acc: 0.5557 - val_loss: 0.6998 - val_acc: 0.5323
```

```
Epoch 27/50
1436/1436 [==============================] - 322s 224ms/step - loss: 0.7456 -
acc: 0.5780 - val_loss: 0.7265 - val_acc: 0.5285
Epoch 28/50
1436/1436 [==============================] - 326s 227ms/step - loss: 0.7717 -
acc: 0.5564 - val_loss: 0.7237 - val_acc: 0.5285
Epoch 29/50
1436/1436 [==============================] - 323s 225ms/step - loss: 0.7640 -
acc: 0.5689 - val_loss: 0.6839 - val_acc: 0.5323
Epoch 30/50
1436/1436 [==============================] - 320s 223ms/step - loss: 0.7428 -
acc: 0.5669 - val_loss: 0.6968 - val_acc: 0.5285
Epoch 31/50
1436/1436 [==============================] - 327s 228ms/step - loss: 0.7340 -
acc: 0.5613 - val_loss: 0.6961 - val_acc: 0.5285
Epoch 32/50
1436/1436 [==============================] - 321s 223ms/step - loss: 0.7547 -
acc: 0.5411 - val_loss: 0.7217 - val_acc: 0.5285
Epoch 33/50
1436/1436 [==============================] - 325s 226ms/step - loss: 0.7379 -
acc: 0.5648 - val_loss: 0.7044 - val_acc: 0.5285
Epoch 34/50
1436/1436 [==============================] - 321s 223ms/step - loss: 0.7167 -
acc: 0.5801 - val_loss: 0.6921 - val_acc: 0.5361
Epoch 35/50
1216/1436 [========================>...] - ETA: 48s - loss: 0.7397 - acc:
```

```
      ␣
 ↪---------------------------------------------------------------------------

       KeyboardInterrupt                         Traceback (most recent call␣
 ↪last)

       <ipython-input-15-ed26bc559431> in <module>
        64     # model.summary()
        65     # Step 4.
    ---> 66     history = model.fit(X_train_inst,Y_true_train_inst , epochs=50,␣
 ↪batch_size=32, validation_data=(X_val_inst,Y_true_val_inst))
        67
        68     plot_history()


       ~\Documents\Conda\lib\site-packages\keras\engine\training.py in␣
 ↪fit(self, x, y, batch_size, epochs, verbose, callbacks, validation_split,␣
 ↪validation_data, shuffle, class_weight, sample_weight, initial_epoch,␣
 ↪steps_per_epoch, validation_steps, **kwargs)
      1037                                         initial_epoch=initial_epoch,
```

```
    1038                                 steps_per_epoch=steps_per_epoch,
 -> 1039                                 validation_steps=validation_steps)
    1040
    1041     def evaluate(self, x=None, y=None,


     ~\Documents\Conda\lib\site-packages\keras\engine\training_arrays.py in fit_loop(model, f, ins, out_labels, batch_size, epochs, verbose, callbacks, val_f, val_ins, shuffle, callback_metrics, initial_epoch, steps_per_epoch, validation_steps)
     197                         ins_batch[i] = ins_batch[i].toarray()
     198
 --> 199                     outs = f(ins_batch)
     200                     outs = to_list(outs)
     201                     for l, o in zip(out_labels, outs):


     ~\Documents\Conda\lib\site-packages\keras\backend\tensorflow_backend.py in __call__(self, inputs)
    2713                     return self._legacy_call(inputs)
    2714
 -> 2715                 return self._call(inputs)
    2716         else:
    2717             if py_any(is_tensor(x) for x in inputs):


     ~\Documents\Conda\lib\site-packages\keras\backend\tensorflow_backend.py in _call(self, inputs)
    2673             fetched = self._callable_fn(*array_vals, run_metadata=self.run_metadata)
    2674         else:
 -> 2675             fetched = self._callable_fn(*array_vals)
    2676         return fetched[:len(self.outputs)]
    2677


     ~\Documents\Conda\lib\site-packages\tensorflow\python\client\session.py in __call__(self, *args, **kwargs)
    1437           ret = tf_session.TF_SessionRunCallable(
    1438               self._session._session, self._handle, args, status,
 -> 1439               run_metadata_ptr)
    1440         if run_metadata:
    1441           proto_data = tf_session.TF_GetBuffer(run_metadata_ptr)
```

```
KeyboardInterrupt:
```

[10]:
```python
import matplotlib.pyplot as plt
from pylab import plot, show, figure, imshow, xlim, ylim, title


def plot_history():
    plt.figure(figsize=(9,4))
    plt.subplot(1,2,1)
    plt.plot(history.history['acc'])
    plt.plot(history.history['val_acc'])
    plt.title('Model accuracy')
    plt.ylabel('Accuracy')
    plt.xlabel('Epoch')
    plt.legend(['Train accuracy', 'Validation accuracy'], loc='upper left')
    plt.subplot(1,2,2)
    plt.plot(history.history['loss'])
    plt.plot(history.history['val_loss'])
    plt.title('Model loss')
    plt.ylabel('Loss')
    plt.xlabel('Epoch')
    plt.legend(['Train loss', 'Validation loss'], loc='upper left')
    plt.show()
```

[ ]:
```python
""""
    # Step 3: simplify the data by averaging over time
    # Instead of having time-varying features, we'll summarize each track by␣
 ↪its mean feature vector over time
    X_train_inst_sklearn = np.mean(X_train_inst, axis=1)
    X_test_inst_sklearn = np.mean(X_test_inst, axis=1)
    X_train_inst_sklearn = X_train_inst_sklearn.astype('float32')
    X_train_inst_sklearn = lb.util.normalize(X_train_inst_sklearn)
"""


np.savez('models.npz',model=)
```