```
from google.colab import drive
drive.mount('/content/drive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_

Enter your authorization code:
··········
Mounted at /content/drive
```

```
from __future__ import print_function
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from time import time
#np.random.seed(1337)  # for reproducibility

from keras.preprocessing import sequence
from keras.models import Sequential
from keras.layers.core import Dense, Dropout, Activation, Flatten
from keras.layers.normalization import BatchNormalization
from keras.layers.convolutional import Convolution1D, MaxPooling1D
from keras.utils import np_utils
from keras.callbacks import TensorBoard


# set parameters:
test_dim = 499
maxlen = 100
nb_filter = 512
filter_length_1 = 10
filter_length_2 = 5
hidden_dims = 750
nb_epoch = 20
nb_classes = 2
split_ratio = 0.15

print('Loading data...')

X = np.load('/content/drive/My Drive/Colab Notebooks/data/numpy_vectors/x_test_mfcc
y = np.load('/content/drive/My Drive/Colab Notebooks/data/numpy_vectors/y_label_500
print(X.shape)
print(y.shape)
```

```
Loading data...
(1498, 499, 13)
(1498,)
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=split_ratio)

xts = X_train.shape
```

```
#X_train = np.reshape(X_train, (xts[0], xts[1], 1))
xtss = X_test.shape
#X_test = np.reshape(X_test, (xtss[0], xtss[1], 1))
yts = y_train.shape
#y_train = np.reshape(y_train, (yts[0], 1))
ytss = y_test.shape
#y_test = np.reshape(y_test, (ytss[0], 1))

print(len(X_train), 'train sequences')
print(len(X_test), 'test sequences')

Y_train = np_utils.to_categorical(y_train, nb_classes)
Y_test = np_utils.to_categorical(y_test, nb_classes)

# print('Pad sequences (samples x time)')
# X_train = sequence.pad_sequences(X_train, maxlen=maxlen)
# X_test = sequence.pad_sequences(X_test, maxlen=maxlen)
# print('X_train shape:', X_train.shape)
# print('X_test shape:', X_test.shape)
```

```
1273 train sequences
225 test sequences
```

```
for batch_size in range(10, 11, 5):
    print('Build model...')
    model = Sequential()

    # we start off with an efficient embedding layer which maps
    # our vocab indices into embedding_dims dimensions
    # model.add(Embedding(max_features, embedding_dims, input_length=maxlen))
    # model.add(Dropout(0.25))

    # we add a Convolution1D, which will learn nb_filter
    # word group filters of size filter_length:
    model.add(Convolution1D(nb_filter=nb_filter,
                            filter_length=filter_length_1,
                            input_shape=(test_dim, 13),
                            border_mode='valid',
                            activation='relu'
                            ))
    # we use standard max pooling (halving the output of the previous layer):
    model.add(BatchNormalization())

    model.add(Convolution1D(nb_filter=nb_filter,
                            filter_length=filter_length_2,
                            border_mode='same',
                            activation='relu'
                            ))

    model.add(BatchNormalization())

    model.add(MaxPooling1D(pool_length=2))
```

```
    model.add(Convolution1D(nb_filter=nb_filter,
                            filter_length=filter_length_2,
                            border_mode='same',
                            activation='relu'
                            ))

    model.add(BatchNormalization())

    model.add(MaxPooling1D(pool_length=2))

    # We flatten the output of the conv layer,
    # so that we can add a vanilla dense layer:
    model.add(Flatten())

    # We add a vanilla hidden layer:
    # model.add(Dense(hidden_dims))
    model.add(Dropout(0.25))
    # model.add(Activation('relu'))

    model.add(Dense(1000))
    model.add(Activation('relu'))
    model.add(Dense(750))
    model.add(Activation('relu'))
    model.add(Dense(50))
    model.add(Activation('relu'))
    # We project onto a single unit output layer, and squash it with a sigmoid:
    model.add(Dense(nb_classes))
    model.add(Activation('softmax'))

    model.compile(loss='binary_crossentropy',
                optimizer='adam', metrics=['accuracy'])

    print("model/split = {} <> batchsize = {}".format(split_ratio, batch_size))
    tensorboard = TensorBoard(log_dir="logs/split_{}_batchsize_{}".format(split_rat

    model.fit(X_train, Y_train, batch_size=batch_size,
            nb_epoch=nb_epoch,  verbose=1, callbacks=[tensorboard]  )

    # model.save('model_hin_tel_38_samples.h5')

    y_preds = model.predict(X_test)
    for i in range(len(y_preds)):
        print(y_preds[i], y_test[i])

    score = model.evaluate(X_test, Y_test, verbose=1)
    print(score)
    print("\n*********************************\n")

# print(classification_report(Y_test, Y_preds))
```

```
Build model...
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:17: UserWarning:
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:25: UserWarning:
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:30: UserWarning:
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:35: UserWarning:
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:40: UserWarning:
model/split = 0.15 <> batchsize = 10
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:68: UserWarning:
Epoch 1/20
1273/1273 [==============================] - 4s 3ms/step - loss: 0.3025 - acc:
Epoch 2/20
1273/1273 [==============================] - 3s 2ms/step - loss: 0.0965 - acc:
Epoch 3/20
1273/1273 [==============================] - 3s 2ms/step - loss: 0.2002 - acc:
Epoch 4/20
1273/1273 [==============================] - 3s 2ms/step - loss: 0.1637 - acc:
Epoch 5/20
1273/1273 [==============================] - 3s 2ms/step - loss: 0.1637 - acc:
Epoch 6/20
1273/1273 [==============================] - 3s 2ms/step - loss: 0.1069 - acc:
Epoch 7/20
1273/1273 [==============================] - 3s 2ms/step - loss: 0.1106 - acc:
Epoch 8/20
1273/1273 [==============================] - 3s 2ms/step - loss: 0.2195 - acc:
Epoch 9/20
1273/1273 [==============================] - 3s 2ms/step - loss: 0.0581 - acc:
Epoch 10/20
1273/1273 [==============================] - 3s 2ms/step - loss: 0.0335 - acc:
Epoch 11/20
1273/1273 [==============================] - 3s 2ms/step - loss: 0.0129 - acc:
Epoch 12/20
1273/1273 [==============================] - 3s 2ms/step - loss: 0.0052 - acc:
Epoch 13/20
1273/1273 [==============================] - 3s 2ms/step - loss: 0.0052 - acc:
Epoch 14/20
1273/1273 [==============================] - 3s 2ms/step - loss: 0.0011 - acc:
Epoch 15/20
1273/1273 [==============================] - 3s 2ms/step - loss: 0.0011 - acc:
Epoch 16/20
1273/1273 [==============================] - 3s 2ms/step - loss: 2.8281e-06 -
Epoch 17/20
1273/1273 [==============================] - 3s 2ms/step - loss: 1.4001e-04 -
Epoch 18/20
1273/1273 [==============================] - 3s 2ms/step - loss: 1.8596e-05 -
Epoch 19/20
1273/1273 [==============================] - 3s 2ms/step - loss: 1.9565e-05 -
Epoch 20/20
1273/1273 [==============================] - 3s 2ms/step - loss: 5.4941e-07 -
[1.000000e+00 8.696429e-11] 0
[1.00000e+00 9.54459e-10] 0
[1.0000000e+00 2.2676615e-10] 0
[0. 1.] 1
[0. 1.] 1
[1.0000000e+00 4.5555018e-09] 0
[1.0000000e+00 8.2831526e-11] 0
```

```
[0. 1.] 1
[0.00161813 0.99838185] 1
[0. 1.] 1
[0. 1.] 1
[0. 1.] 1
[0. 1.] 1
[0. 1.] 1
[9.9999976e-01 2.0641473e-07] 0
[0. 1.] 1
[1.0000000e+00 2.1830193e-10] 0
[1.0000000e+00 3.6329167e-10] 0
[1.000000e+00 9.098621e-10] 0
[1.0000000e+00 2.6457633e-11] 0
[1.0000000e+00 8.7732205e-10] 0
[1.000000e+00 7.220523e-10] 0
[1.0000000e+00 1.7962258e-11] 0
[1.000000e+00 4.775263e-09] 0
[0. 1.] 1
[0. 1.] 1
[0. 1.] 1
[1.000000e+00 8.638773e-13] 0
[1.0000000e+00 2.1075184e-09] 0
[0. 1.] 1
[0. 1.] 1
[1.0000000e+00 1.0696566e-10] 0
[0. 1.] 1
[0. 1.] 1
[0. 1.] 1
[1.0000000e+00 1.9536676e-11] 0
[1.000000e+00 7.493106e-10] 0
[1.000000e+00 6.170298e-09] 0
[2.8382912e-36 1.0000000e+00] 1
[0. 1.] 1
[1.0000000e+00 9.1052005e-10] 0
[1.0000000e+00 2.2071475e-10] 0
[0. 1.] 1
[1.0000000e+00 1.7616233e-09] 0
[1.0000000e+00 4.3747853e-10] 0
[0. 1.] 1
[1.000000e+00 5.004872e-09] 0
[1.0000000e+00 2.7773425e-10] 0
[9.9998450e-01 1.5487256e-05] 0
[1.000000e+00 4.453353e-09] 0
[0. 1.] 1
[1.0000000e+00 5.3168197e-08] 0
[9.999999e-01 7.113794e-08] 0
[9.999999e-01 7.488327e-08] 0
[0. 1.] 1
[0. 1.] 1
[1.00000e+00 5.48399e-09] 0
[0. 1.] 1
[1.000000e+00 7.709662e-12] 0
[0. 1.] 1
[0. 1.] 1
[0. 1.] 1
```