# 真伪语音鉴别实验报告

南京林业大学 马维龙

1. **实验分析**
2. 本实验需要根据train文件夹下的.flac文件以及train.txt中的结果对模型进行训练，最终实现识别出一段语音的真伪。
3. 为了可以对一段语音进行训练，故可以将此文件的特征值提取出来，用特征值训练模型。
4. 由于本实验的目标为识别出真伪语音，即有两种评判标准，故可以将文件中的真伪语音分离出来，分别训练两个模型。
5. 既然已经训练出来了两个模型，可以用两个模型对同一段语音分别打分，分高的一段则判断文件最终何种类型的语音。
6. **实现步骤**
7. 特征值提取
   1. 用librosa根据文件路径读取语音文件音频时间序列，采样率等信息；
   2. 根据获得的信息以及pdf中提供的参数，通过mfcc获取特征值；
   3. 返回修正后的特征值。

**代码实现：**

def get\_Mfcc(path):

    num\_ceps = 13

    low\_freq = 0

    high\_freq = 2000

    nfilts = 24

    nfft = 512

    dct\_type = 2

    use\_energy = False

    lifter = 5

    normalize = False

    sig, fs = librosa.load(path, sr=None)

    mfcc\_feature = mfcc(sig=sig,

                 fs=fs,

                 num\_ceps=num\_ceps,

                 nfilts=nfilts,

                 nfft=nfft,

                 low\_freq=low\_freq,

                 high\_freq=high\_freq,

                 dct\_type=dct\_type,

                 use\_energy=use\_energy,

                 lifter=lifter,

                 normalize=normalize)

#     mfcc\_feature = librosa.feature.mfcc(sig, fs, n\_mfcc=13)

    mfcc\_feature = preprocessing.scale(mfcc\_feature)

    deltas = delta(mfcc\_feature,2)

    double\_deltas = delta(deltas,2)

    combined = np.hstack((mfcc\_feature,deltas,double\_deltas))

    return combined

1. 模型训练
   1. 根据train.txt中的文件标签将音频文件分类；
   2. 根据标签分两次训练模型；
   3. 每次训练同一种标签的音频文件，对每个文件调用get\_Mfcc获取特征值；
   4. 将特征值整合后用高斯混合模型进行训练。

**代码实现：**

nowname = "bonafide"

source="zju\_deepfake\\train\\flac\\"

dest="zju\_deepfake\\"

features=np.asarray(())

spoof\_name = open("zju\_deepfake\\" + nowname + ".txt")

while(True):

    # 读取某次训练中的每个音频

    line = spoof\_name.readline()

    if not line:

        break

    path = source + line[:12] + ".flac"

#     print(path)

    vector = get\_Mfcc(path)

    if features.size == 0:

        features = vector

    else:

        features = np.vstack((features, vector))

print(features)

#GMM模型

# print(minv)

# print("!")

gmm = GaussianMixture(n\_components=8,covariance\_type='diag',max\_iter=200,n\_init=3)

print(gmm)

gmm.fit(features)

picklefile= nowname + ".gmm"

print(dest+picklefile)

#写模型

pickle.dump(gmm,open(dest+picklefile,'wb'))

1. 判断真伪
   1. 读取音频文件计算特征值；
   2. 根据训练处的两个模型对特征值进行打分；
   3. 分数高的为当前音频的真伪；
   4. 将文件名和结果标签写入结果文件。

spoof\_model\_path = "zju\_deepfake\\spoof.gmm"

bonafide\_model\_path = "zju\_deepfake\\bonafide.gmm"

# 载入训练好的模型

spoof\_gmm = pickle.load(open(spoof\_model\_path, 'rb'))

bonafide\_gmm = pickle.load(open(bonafide\_model\_path, 'rb'))

def identify(vector):

    spoof\_scores = np.array(spoof\_gmm.score(vector))

    spoof\_score = spoof\_scores.sum()

    # male hypothesis scoring

    bonafide\_scores = np.array(bonafide\_gmm.score(vector))

    bonafide\_score = bonafide\_scores.sum()

#     print("%10s %5s %1s" % ("+ SPOOF SCORE",":", str(round(is\_spoof, 3))))

#     print("%10s %7s %1s" % ("+ BONAFIDE SCORE", ":", str(round(is\_bonafide,3))))

    if spoof\_score > bonafide\_score:

        winner = "spoof"

    else:

        winner = "bonafide"

    return winner

nowfile = "eval2"

filepath = "zju\_deepfake\\" + nowfile + "\\flac\\"

ans\_file = "zju\_deepfake\\" + nowfile + "\_ans.txt"

ans\_out = open(ans\_file, "w")

prediction = "spoof"

for root, dirs, files in os.walk(filepath):

    for f in files:

        vector = get\_Mfcc(filepath + f)

        prediction = identify(vector)

        ans\_out.write(f[:12] + " " + prediction + "\n")

1. **自测结果**

根据提供的dev中的音频和结果，正确率约为82.44%