Audio-visual speech processing project using an idea binary mask: README

*Arun Palghat Udayashankar*

*p.u.arun@gmail.com*

July 29, 2014

Note: In order to copy the code to another machine please put the Work/AV\_project under C: drive with the same subdirectory ordering so that the code can be executed as is.

DATA

Data directory root:  **C:\Work\AV\_project\data**

Audio: \*.wav files

Video: \*.csv files containing marker data

Naming convention for sound files: sentenceNumber\_listNumber\_videoFPS.wav

Naming convention for marker files: sentenceNumber-listNumber.csv

CODE

Code root directory: The root directory contains a function called TrainClassifier.m. Call this function in the following manner:

*I) MASK and Features:*

**C:\Work\AV\_project\code\BinaryMask\idbm**

***makeFeatures(SNR) -> extractMASKnFeatures() -> designMelfilterbank ; extractFeatures***

In the above function call SNR represents the signal to noise ratio. This function will save the features and oracle to the preconfigured path in to be given at the end of the function.

*extractMASKnFeatures()*:

This function adds in the noise, does sub band filtering and prepares files for feature and mask extraction. The type of noise is specified in this function.

*extractFeatures()*:

This function does the actual feature extraction from the envelope using FFT and triangular filtering

At the end of step I you are left with the features and oracle for the desired SNR and noise type (specified in *extractMASKnFeatures*)

*Addnoise: add in noise to the speech file*

*addVidfeatures: add in additional video features*

*designMelfilterbank: design an FIR filterbank with mel spacing.*

*extractFeatures: extract the first 15 audio features and MASK for a given band*

*extractMASKnFeatures: extract the features 30-45 for audio and MASK for all bands and put the MASK and features into a giant array.*

*Frames2vec: convert frames to a vector using window functions*

*getFeaturesFromMarkers: extract the video features from csv marker files*

*hline: draw a horizonatal line*

*vline: draw a vertical line*

*makeFeatures: function to be called with passing SNR as argument to make features. The default noise type is white noise. In roder to use another type of noise change it under extractMASKnFeatures::preparespeechfile().*

*Myspectrogram: given the sampling frequency and audio plot a spectrogram*

*Reconstructaudio: given the mask and audio file apply the mask to make a new audio file*

*Segsnr: compute SNR*

*SSBoll79: our initial recordings were noisy and so all speech files were passes through*

*Trifbank: triangular filter bank*

*Vec2frames: vector to frames*

II) LinearRegression:

**C:\Work\AV\_project\code\BinaryMask\idbm\LinearRegression**

**runRegression() -> checkPerformance()**

This function will return the regression coefficients by calling checkPerformance. Keep track of the path of features and the model output.

*Checkperformance: this function will do the core of the model fitting process*

*Finnonzero: this function will find non-zero features by scanning the beta coefficients after model fitting*

*Hline: horizontal line*

*Myspectrogram: plot spectrogram*

*pickThreshold: this function help you pick a threshold visually*

*plotPercentCorrect: self explanatory*

*runRegression: initial pathas and run regression*

*testperformance: use the fitted model to evaluate your results.*

*Testperscript: run this to make sound files after fitting linear model*

*Visualizeclassifiers: visualize the percent right of the classifiers.*

*Visualthrehsold: pick the threshold visually.*

*Vline: vertical line.*

III) Make sound files using linear model:

**C:\Work\AV\_project\code\BinaryMask\idbm\LinearRegression**

**testPerscript() -> testPerformance->reconstructaudio()**

This function will loop through all the files specified by r (index) and it will apply the MASK returned by the Video, Audio and Audio –Video models. In addition it will also save the noisy files and files to which the IDBM has been applied. Make sure to configure path to the model created in step II. All results are saved to the /results directory, make sure to move them to the directory of your choice.

IV) Gaussian mixture model:

**C:\Work\AV\_project\code\BinaryMask\idbm\GMMBayes**

This step makes use of the Gaussian mixture model – Bayes toolbox from: <http://www.it.lut.fi/project/gmmbayes/>. Kindly review the link and run the toolbox demos before proceeding to gain a basic understanding of the model and its implementation. The specific GMM-Bayes we use in our code is Greedy GMM. In order to fit the GMM model we do the following: run the script **loadOracleFeatures.m -> featureAnalysisGMMBayes** . Make sure that the directory variables are correctly configured for the features, linear model and oracle. The second script featureAnalysisGMMBayes will call the functions in the GMMbayes toolbox. It may be convenient to save the linear model and the non-linear model in the same directory. The non-linear model fitting process takes into account only those features that were not zeroed out in the linear model fitting process. Hence we call it here.

*applyGMMBayes: run this to apply the fitted GMMBayes model to your sound file.*

*changeProbability: just atets function to the effect of changing probability on percent correct.*

*runGMMBayes: this script will get the model fitting process started.*

*Plotcmat: run this function by passing in the cmat array after model fitting to visualize percent correct*

*testGMMBclassifier: this does the heavylifting of making the sound files by applying the model.*

V) Make sound files with the GMM:

**C:\Work\AV\_project\code\BinaryMask\idbm\GMMBayes**

**applyGMMBayes->testGMMBayesClassifier**

Configure path to fitted non-linear model. Use the indices of sentences to be processed (r). All output files in applyGMMBayes. Configure path to linear model in testGMMBayes. Run the script and the output files are written to the results folder.

VI) Test GUI:

**C:\Work\AV\_project\code\BinaryMask\idbm\TestGUI**

**speechTest(.m/fig):** this was done using GUIDE, a matlab tool. The individual functions are commented in the code. Before starting the test process all 400 sntences under the different conditions like idbm, noise, audio, video and audio-video and store them in a subdirectory. Condifgure this path in the start button push event and then start the test.

**Separatefiles:** this will pick only indices of those files that were not:

1. Degraded by SSBoll79 to remove initial recording noise
2. Have no context

**Restdata:** indices of files that were not degraded by SSBoll79.

**Calibration:** in order to calibrate your master volume.