LIA_SPKDET

Package documentation

Crédits

This documentation is a user guide:

Editeur / Editor: Eric Charton – eric.charton@univ-avignon.fr
Remarque / Comments :
This edition of the documentation have not been modified. It's dedicated to the 1.3 of LIA_SPKDET Package. It will be rewrited for the 2.0 version

Cette version reprends et enrichit la version précédente / this version follow this edition:
1.0
Auteur / Writer : LIA_Team

Sommaire

1. INTRODUCTION TO LIA_RAL

This package aims at providing Automatic Speaker Detection related programs based on the ALIZE toolkit. It contains two task-specific

sub-packages : LIA_SpkDet related to Speaker Detection and LIA_Seg related to Speaker Diarization and acoustic segmentation. A library

containing useful methods is provided in LIA_SpkTools as well as useful, uncategorized programs in LIA_Utils. This documentation is

dedicated to the Speaker Detection package, LIA_SpkDet

2. LIA_SPKDET

2.1. General Options

Options described below are shared by most of the programs of LIA_RAL. The reader is invited to get deeper in the ALIZE way of

working as the meaning of some options (marked with an X in the ALIZE column) is closely related to the philosophy of ALIZE.

2.1.1. Model related Options

Name	Example	Description	ALIZE
distribType	GD	Specify the type of distribution	X
mixtureDistribCount	10	Specify the number of gaussian distributions in the mixture	X
loadMixtureFileFormat	RAW	Specify the loading / saving format of the distribution	X
saveMixtureFileFormat	XML		
loadMixtureFileExtension	.gmm	Specify the extension in order to save / to load a model	X
saveMixtureFileExtension	""		X
mixtureFilesPath	./	Specify the path where to load and save mixtures	X

2.1.2. Feature related Options

Name	Example	Description	ALIZE
loadFeatureFileExtension	.prm	Specify the extension in order to save / to load a feature file	X
saveFeatureFileExtension	.norm.prm		X
saveFeatureFileSPro3DataKind	FBCEPSTRA	Specify the saving SPro 3 format of the file	X
loadFeatureFileFormat	SPRO3	""	X
saveFeatureFileFormat	HTK	Specify the loading/saving format of a feature file	X

featureFilesPath ./ Specify the path where to load and save feature files X

vectSize 32 Specify the dimension of vectors in the feature files X

featureServerMode FEATURE_WRITABLE Specify the read/write mode of the featureServer (read-only by X default)

featureServerMemAlloc 100000 Specify the memory to allocate for the ALIZE buffer X

2.1.3. Segment, Labels, Clusters related Options

Name Example Description ALIZE

labelFilesPath ./ Specifiy the path where to load and save label files X

labelSelectedFrames male Specify the label to work with

useIdForSelectedFrame The label selected is the same as the name found in the input feature list

addDefaultLabel true When no label file is found, add a default label to all features

defaultLabel male Specify this default label

frameLength 0.01 When working in segmental mode, specify the ratio between the time unit found in the label files and the frame-based time unit

2.1.4. Misc. Option

Name Example Description ALIZE

minLLK -200 Specify minimum and maximum likelihood values X

maxLLK 200 X

bigEndian false Tune this option depending on your O.S. X

featureServerBufferSize ALL_FEATURES Algorithm Buffer X

Alize_featureBuffer 100 Memory Buffer X

debug true Enable debug mode

verbose true Enable verbose mode

2.2. EnergyDetector

2.2.1. Features

EnergyDetector aims at analyzing the energy component of input features by producing an output label file labeling features with

highest energy. This is a typical speech/non-speech detection. The output label files get their label from the labelOutputFrames option. The

process of selecting files is done via two parameters :

– mixtureDistribCount is used as the frames are selecting by the analysis of a GMM learned on them (use 2 or 3)

– alpha is the percentage of frames selected in the central gaussian.

2.2.2. Compulsory options

Name	Example	Description
inputFeatureFileName	test1	Name of the features to work with, can be a list with a .lst extension
labelOutputFrames	speech	Label affected to selected frames
saveLabelFileExtension	.enr.lbl	Label file to save extension
nbTrainIt	10	Number of EM iterations to estimate energy distribution
varianceFlooring	0.5	variance control parameters
varianceCeiling	10	
alpha	0.25	Percentage of frames selected in the central gaussian, idle when mixtureDistribCount==2
featureServerMask	16	In this case, energy detection is done on a single dimension of the input vectors
baggedFrameProbabilityInit	0.1	Specify the ratio between selected frames and the total number of frames used for initialisation, in this case 1 out of ten frames are took in account

2.3. NormFeat

2.3.1. Features

NormFeat aims at processing input speech related features (MFCC, LFCC, ...) by applying any kind of normalization. Its main functio-

nality is to map the distribution of input features on another one, e.g. when one wants their features to fit a gaussian distribution of 0 -mean

and 1 -variance.

2.3.2. Compulsory options

Name	Example	Description
mode	norm	Specify the mode of NormFeat : norm is for feature normalization on a $\mathcal{N}(0,1)$ distribution, info gives mean and std for each component, featMap computes the feature Mapping

inputFeatureFileName	./lst/foo.lst	Name of the features to work with, can be a list with a .lst extension
segmentalMode	false	When working in segmental Mode, stats and normalization are computed for each segment independently
writeAllFeatures	true	When set to true, all features on the original file are written, when set to false, only normalized features are written
indepModel	CI	These two options are only compulsory in featMap mode. The first gives the channel independent model, being the target distribution to map the features to, the second gives the channel dependent model of the features one wants to map
depModel	CD	

2.3.3. Example

An input feature file is obtained from a signal with 2 different speakers of two different genders. The objective is to make the distribution of the features belonging to the male fit $\mathcal{N}(0,1)$. The corresponding label file might look like this :

0 120.00 male

120.00 154.00 female

154.00 200 male

Two steps are necessary to achieve this goal :

– The first step takes into account only the male features, given by the labelSelectedFrame==male option, compute the mean and covariance on all segments belonging to this label (thanks to the –segmentalMode option tuned to false), then make them fit the normal distribution. The remaining female features are written thanks to the writeAllFeatures=true option

– The second step only consists of repeating the first step while changing the labelSelectedFrame option to female

Command Line : ./NormFeat.exe –config NormFeat.cfg

2.3.4. Feature Mapping

FeatureMapping is an extension of mapping the input feature distribution to a distribution different from $\mathcal{N}(0,1)$. The aim of this mode is to map the current input feature to the best gaussian of a channel independent model. The knowledge of the belonging of the current feature to a channel dependent model is stored in a label file. Another program is necessary to create this latter. One will have to repeat the following command line as many as the number of existing channel dependent models.

Command Line : NormFeat.exe –config cfg/Normfeat.cfg –mode featMap –indepModel ./World –depModel <./Cellular ./Landline ./Court>

2.4. TrainWorld

2.4.1. Features

TrainWorld is a generic application of learning a Gaussian Mixture Model via the EM (Expectation - Maximization) algorithm.

Initialization is done by picking features randomly in the inputFeatureFilename (this parameter can be a list of feature files) according to

the probability given by baggedFrameProbabilityInit.

Input feature dimension and target GMM model's one do not have to match thanks to featureServerMask.

As well as other programs of this toolkit, TrainWorld can work in segmental mode. The frameLength option has to be filled when the given

label file provides segmental time unit in an other unit than frame.

Two kinds of EM iterations are made during the learning process. The first one, which amount is given by nbTrainIt picks features randomly

in the inputFeatureFilename according to the probability given by baggedFrameProbability.

The learning process finishes by iterations given by the nbTrainFinalIt option, for which all frames are taken into account.

2.4.2. Compulsory options

Name	Example	Description
baggedFrameProbabilityInit	0.1	Specify the ratio between selected frames and the total number of frames used for initialization
baggedFrameProbability	0.01	Specify the ratio between selected frames and the total number of frames used for training
normalizeModel	false	Apply a $1,0(x)$ mapping at the end of the learning process
inputFeatureFilename	seg_app.lst	Name of the features to work with, can be a list with a .lst extension
fileInit	false	Specify if a model is available to skip the initialization step
inputWorldFilename	xxx	
initVarianceCeiling	10	
initVarianceFlooring	1	Variance control parameters, linearly decreasing or increasing from init to final
finalVarianceFlooring	0.5	

finalVarianceCeiling 5

nbTrainIt 2 Number of EM iterations related to baggedFrameProbability

nbTrainFinalIt 1 Number of final EM iterations with no baggedFrameProbability

outputWorldFilename wld Name of the resulting file model

2.5. TrainTarget

2.5.1. Features

TrainTarget is a generic application to achieve the adaptation process of a model on data. In the ASR field, it is commonly used to train

target speakers by adapting a world model via a Maximum A Posteriori method. Several MAP methods have been implemented :

– MAPConst : the random variable to estimate is computed by a linear combination of its value in the world model and its value

obtained by an EM algorithm on the data. The weights of this combination are provided by the option alpha (for the world model

and 1 for the client model). These methods adapt a background GMM model on a specific set of data by duplicating the world

model and changing its means. Indeed, no variance neither weights adaptation are implemented.

– MAPOccDep : the random variable to estimate is computed by a linear combination of its value in the world model and its value

obtained by an EM algorithm on the data. This method takes into account the a posteriori probability n for each gaussian. The

n n

weights of this combination are provided by the option MAPRegFactor r (n+r for the world model and 1 n+r for the client

model).

2.5.2. Compulsory options

Name	Example	Description
baggedFrameProbability	0.01	Specify the ratio between selected frames and the total number of frames used for adaptation
nbTrainIt	5	Number of EM iterations related to baggedFrameProbability
nbTrainFinalIt	1	Number of final EM iterations with no baggedFrameProbability
normalizeModel	false	Apply a $1,0(x)$ mapping at the end of the learning process
targetIdList	ndx	The input list : model on the first column, input feature file-names on others

inputWorldFilename	wld	The a priori model
MAPAlgo	MAPOccDep	Specify the adaptation method to use : MAPOccDep or MAP-Const
MAPRegFactor	10	parameter used by the MAPOccDep adaptation technique
alpha	0.75	parameter used by the MAPAlgo adaptation technique

2.6. ComputeTest

2.6.1. Features

ComputeTest is a generic application aiming at giving a score for a test segment given a model. Multiple test segments with multiple models are supported by this program. ComputeTest implements the winning component scoring method. This can be achieved by filling the computeLLKWithTopDistribsoption with COMPLETE and topDistribsCount by the desired number of components to use. Instead of giving a single score related to one file and one model, ComputeTest can give a score for each segment given in the input label file, this done by filling the option segmentalMode with segmentLLR.

2.6.2. Compulsory options

Name	Example	Description
segmentalMode	segmentLLR	if segmentLLR, gives a score for each input segment in the label file
topDistribsCount	10	Number of gaussians used to compute the score
computeLLKWithTopDistribs	COMPLETE	See ALIZE documentation for this
ndxFileName	ndx	input file : test segment on the first column, models to test with on the others
worldModelName	wld	background model
gender	M	Gender of the ndx file
outputFile	test1.res	Resulting score file

2.7. ComputeNorm

2.7.1. Features

ComputeNorm aims at applying normalization technique on speaker detection scores. The various techniques implemented are znorm, tnorm, and ztnorm. It has to be noted that the impostor trials and related impostor scores, necessary for these normalization techniques, are not performed by this program (ComputeTest has to be used for this task).

During the normalization process, it is possible to use all the impostor scores (selectT ypenoSelect) or to select part of them, either by

retrieving the N best ones (selectT ypeselectN BestByT estSegment) (the N impostors for which the likelihoods between their models

and the test signal are the greatest ones) or by reading impostor cohort from a given client dependent list (selectT ypeselectT arget-

DependentCohortInF ile). For the latter, the form of the client dependent list has to be one impostor id per line (the impostor id being

related to those used in the Z/T/ZTnorm score files). Note : ComputeNorm program currently works only on non segmental input file NIST.

2.7.2. Compulsory options

Name	Example	Description
outputFile	test1.nist	nist normalized score file
normType	znorm tnorm ztnorm	Normalization technique to apply
testNistFile	nist	input file : client score file in NIST format (V. 2003)
tnormNistFile	nist	input file : tnorm score file in NIST format (V. 2003) read if normType=tnorm or ztnorm
znormNistFile	nist	input file : znorm score file in NIST format (V. 2003) read if normType=znorm or ztnorm
ztnormNistFile	nist	input file : ztnorm score file in NIST format (V. 2003) read if normType=ztnorm
selectType	noSelect	specify how a set of impostor score may be selected (see description of the program) selectNBestByTestSegment selectTargetDependentCohortInFile
cohortFilePath	path	path where client dependent cohort files may be read
cohortFileExt	extension	extension to use with dependent cohort files (the name of the file being the id of the client
znormCohortNb	integer	size (N) of znorm impostor cohort for the selectNBestByTest-Segment selection technique
tnormCohortNb	integer	size (N) of tnorm impostor cohort for the selectNBestByTest-Segment selection technique
maxSegNb	integer	Maximum number of segment tests to be

authorized

maxScoreDistribNb	integer	Maximum number of impostor scores to be authorized
maxIdNb	integer	Maximum number of client id to be authorized
maxSegNb	integer	Maximum number of Segment tests to be authorized
outputFile	test1.res	Resulting score file

1.Chapitre