NAME

CtuCopy − universal feature extractor and speech enhancer.

SYNOPSIS

ctucopy [*options*]

DESCRIPTION

CtuCopy is a command line tool implementing speech enhancement and feature extraction algorithms. It is similar to HCopy tool from Cambridge's HTK and is partially compatible with it. CtuCopy acts as a filter with speech waveform file(s) at the input and either a speech waveform file(s) or feature file(s) at the output.

CtuCopy implements several speech enhancing methods based on spectral subtraction. Extended Spectral Subtraction − exten [Sovka et al., EUSIPCO'96] combines Wiener filtering and spectral subtraction with no need of a voice activity detector (VAD). Other methods are based on spectral subtraction with VAD which can be either external (data read from file) or internal (Burg's cepstral detector). The noise suppression methods can be applied either directly to speech spectra or to a spectra filtered by a bank of filters.

Bank of filters offers a set of frequency scales (melodic, Bark, expolog, linear) and various filter shapes (triangular, rectangular, trapezoidal same as time-domain IIR based implementation) which can be combined to form standard and user-defined filter banks.

In feature extraction mode a number of common features can be extracted from original or enhanced speech, e.g. LPC, PLP, MFCC or magnitude spectrum. Current version also implements TRAP-DCT features. Features can be saved either in HTK, pfile, or KALDI format.

OPTIONS

CtuCopy recognizes these options:

INPUT/OUTPUT:

−format_in <*fmt*> Input speech file format.
<*fmt*> = raw − sequence of 16b integers
alaw − sequence of 8b integers, A−law encoded
mulaw − sequence of 8b integers, mu−law encoded
wave − MS wave file, PCM 16b, mono

−format_out <*fmt*>

Output file format. This option also determines whether feature extraction should be performed or not. Output format can either be a speech file (raw, wave) or feature file (htk, pfile). In pfile case all input files are saved to the only one output file which must be specified by *filename*, see below.
<*fmt*> = raw − sequence of 16b integers
wave − MS wave file, PCM 16b, mono
htk − HTK feature file
pfile=<*filename*> − ICSI pfile.
ark=<*filename*> − KALDI ark,scp format.

−endian_in <little|big>

Input byte order. Little endian means LSB..MSB, big endian the opposite. Example: number 1 stored in two bytes looks like 10 for little, and 01 for big endian. It does not apply to MS wave files, which are read always in machine native format (displayed by CtuCopy when invoked with −v).

−endian_out <little|big>

Output byte order (see -endian_in). Applies only to raw and htk outputs. Otherwise native order is used.

−online_in
Input will be read from standard input. This option requires single file mode at the output (output either to standard output or to a single file specified with −o option). The input file size is generally not known at the beginning, therefore for MS wave input format the size in file's header is ignored and the input is read until EOF.

−online_out
Similar to −online_in. Requires single file mode at the input. Online output is only applicable to output raw and htk formats which are then saved with no header.

−S <*filename*>
Specifies list of files to be processed. The *filename* structure is one input and output filenames per line. Input and output are separated by one or more tabs or spaces. When format_out is pfile, then output string is discarded and a filename specified in −format_out option is used instead.

−i <*filename*>
Single input file. This option overrides the −S option. For debugging and testing purposes.

−o <*filename*>
Single output file. See −i option.

−preem <*float*>
Preemphasis factor to be applied to the input speech. Preemphasis formula is y[k] = x[k] − *preem**x[k−1]. Value has to be in the range <0.0−1.0) and 0 means no preemphasis.

−fs <*int*>
Sampling frequency in Hz. This option must be explicitly set as it is used for computation of window sizes. MS wave files at the input are checked to match this value before processing.

−dither <*float*>
Dithering constant. A small dither can be added to the input signal. Value 0.0 means no dither, 1.0 means random number in range <−1,+1> at uniform distribution. 1.0 is default.

−remove_dc <on|off>
Remove DC from every signal frame (done after preemphasis and Hamming window).

SEGMENTATION:

−w <*float*>
Window size in milliseconds. The true window length in samples is internally computed from sampling frequency.

−s <*float*>
Window shift size in milliseconds. See −w option.
NOTE: When output is speech, it is recommended to choose 50% or 75% overlap (w/s = 2 or w/s = 4), otherwise there might be a significant ripple in output signal envelope due to OLA. When invoked with −v option, CtuCopy computes and prints out the maximum ripple for chosen values in % before computation.

FILTER BANK:

−fb_scale <*scale*>
Type of the frequency scale warping. By default CtuCopy designes a warped version of the original frequency axis and further designes a set of filters which are equidistant on the warped scale.
<*scale*> = mel − melodic scale (used in MFCC)
bark − Bark scale (used in original PLP)
lin − linear scale (no change)
expolog − special scale originally used for experiments with Lombard speech by Hansen

−fb_shape <*shape*>
Shape of filters. By default every shape has a predefined overlap of subsequent filters (see below). When a different overlap is needed, the filter bank needs to be designed completely by the user

using full −fb_definition specification (see −fb_definition).
<*shape*> = triang − triangular shape with 50% overlap
rect − rectangular shape with no overlap
trapez − trapezoidal shape used in PLP analysis.
NOTE: Trapezoidal shape can only be used in PLP filter bank. As this special shape is related to critical band theory, user cannot choose number of filters in the bank, neither their positions. Thus, when trapez shape is chosen, options −fb_scale and −fb_definition are ignored and a special filter bank is designed.

NOTE 2: See BUGS section for more information on rectangular shape.

−fb_norm <on|off>
Normalize filters to have unit area. Filters that are spread uniformly on the warped frequency scale cover different number of frequency bins of the underlying linear frequency scale as given by FFT. When white noise enters the filter bank, it is no more white after filtering. This option compensates for this effect.

−fb_power <on|off>
Use FFT power spectrum at the input to the filter bank instead of magnitude spectrum.

−fb_eqld <on|off>
Apply a simplified Equal Loudness Curve to the filter bank, prioritizing central frequencies (auditory−like operation). Note that Equal Loudness is applied after possible normalization of filter area (see −fb_norm).

−fb_inld <on|off>
Apply Intensity Loudness Power Law (auditory−like operation). It is a nonlinear operation applied independently to every frequency bin after filtering the spectrum with a filter bank. It is represented by a cube square root (compression of dynamics).

−fb_definition <*string*>
Defines location of all filters in the filter bank. The filter bank consists of a number of filters that are fully described by <*string*>. Filters can be split to several subsets. Every subset is represented by one *token*. The *token* describes a set of filters located equidistantly on the warped frequency axis (specified by −fb_scale option); it specifies start and end positions of the subset in Hz units and also how many filters should be placed in that location. See more explanation below *string* definition.

<*string*> = *token*[,*token*]...
token = [[*X*−*Y*Hz:]*K*−*L*/]*N*filters
X,*Y* ... frequency limits of this filter subset in Hz. The first filter starts at *X* Hz, the last filter ends at *Y* Hz, inclusively. If omitted, defaults to 0−fs/2.
N ... number of equidistant filters in the subset
K,*L* ... optionally specifies a subset of the above filters (indexed from 1). If not specified, defaults to 1−*N*.

Most of the time only one token is used with the default settings. For example, a MFCC filter bank with 25 filters can be specified with "−fb_scale mel −fb_definition 25filters". It says "Place 25 equidistant filters on melodic scale from 0Hz to (fs/2)Hz.". More examples can be found in examples section.
NOTE: There must NOT be any whitespace in <*string*>.

NOISE REDUCTION:

−nr_mode <*mode*>
Algorithm for additive noise suppression. These algorithms are accompanied by a number of options (see below) which need to be properly set for a good performance. Is is recommended to use presets (see −preset option) and then possibly modify the settings.

<*mode*> = exten − Extended Spectral Subtraction
hwss − half−wave rectified spectral subtraction with VAD
fwss − full−wave rectified spectral subtraction with VAD
2fwss − 2−pass full−wave rectified s.s. with VAD
none − no noise reduction

−nr_p <*double*>
Integration constant for spectral subtraction. It affects the smoothness of noise/speech estimation via the formula "NewEstimate = *p* * OldEstimate + (1 − *p*) * Observation". Should be somewhere below 1.0.

−nr_q <*double*>
VAD threshold used in internal Burg's cepstral detector.

−nr_a <*double*>
Magnitude spectrum power constant for spectral subtraction. Defines a domain where SS takes place. Normally either 1 (magnitude) or 2 (power) spectrum. However, any floating number can be used with some performance drawback. Note that this option does not affect the domain at the input to the filter bank, see option -fb_power for more.

−nr_b <*double*>
Noise oversubtraction factor (applies to NR modes hwss and fwss). It is a multiplicative factor applied to the noise estimate before it is subtracted from the input speech−and−noise mixture.

−nr_initsegs <*int*>
Number of initial frames in every input file that are guaranteed not to contain any speech. It is used for initial estimate of noise in algorithms hwss, fwss and 2fwss.

−vad <*string*>
Specification of Voice Activity Detector (VAD). Applies to NR modes hwss, fwss and 2fwss.

<*string*> = burg − built−in Burg's cepstral detector
file=<*filename*> − read VAD information from *filename*.

File <*filename*> is a simple text file containing a sequence of characters 0 or 1, one char for every input signal frame. 0 means no speech in the frame , 1 means the opposite. For more files at the input, there is only one VAD file containing as many characters as there are overall frames at the input.

−rasta <*filename*>
*** *NOT SUPPORTED IN VERSION 3.0* ***
Perform RASTA filtering with impulse responses loaded from the file *filename*.

−nr_when <beforeFB,afterFB>
When used as feature extractor, this specifies whether noise reduction should be done at the output of filter bank (afterFB) or rather before applying the filter bank (beforeFB).

FEATURE EXTRACTION:

−fea_kind <*kind*>
Specifies the type of output features and thus feature extraction algorithm.

<*kind*> = spec − use directly the output of the filter bank as features
logspec − logarithm of spec
lpa − Linear Predictive Coefficients. It means to take power of filter bank outputs (unless −fb_inld is on), take iDFT, and run Levinson−Durbin.
lpc − Linear Predictive Cepstrum. The same as for lpa plus the recursion from LP coeffs to cepstral coeffs. Used e.g. in PLP.
dctc − Cepstrum obtained with DCT. It means taking log of filter bank outputs and projecting to DCT bases. Used e.g. in MFCC.

trapdct,<int>,<int> − TRAP-DCT cepstral coeffs calculated from log mel spectra. First arg is TRAP length in frames, second arg is the number of the first DCT coefficients to save per band. Per-frame log mels are buffered and on their temporal trajectories (the current frame plus a symmetrical window) a Hamming window is applied followed by DCT transform. This is done independently in all frequency bands.

td-iir-mfcc − MFCC-like features which are based on the bank of IIR filters with frequency responses close to standard triangular MFCC filter-bank. Input signal is filtered by the set of distinct IIR filters in the time domain and band power spectrum is then obtained by the computation of short-time power of each particular band signal from the time window of the length specified by -w and -s optionn. Next steps (logarithm and DCT application) are same as for the standard approach of MFCC computation.

-fea_delta <*kind*>
Specifies the type of output differential feats.

<*kind*> = d − compute dynamic diff. coefficients (delta)
d_a − compute dynamic, acceleration diff. coefficients (delta-delta-delta)
d_a_t − compute dynamic, acceleration, third diff. coefficients (delta-delta-delta-delta-delta-delta)

-fea_trap <*int*>
The creation of context information from features defined by -fea_kind. The context information is caught using context window of specified length created from several neighbouring short-time frames (stacked features). The context window size is specified by the integer value (number of frames; the value 5 => cw=5+5+1, i.e. the length of context widnow equal to 11)

−fea_lporder <*int*>
Order of Linear Predictive model (when applicable).

−fea_ncepcoeffs <*int*>
When applicable, number of cepstral coefficients included in the feature vector. It does not include c0, which has its own option −fea_c0.

−fea_c0 <on|off>
Add zeroth cepstral coefficient to the feature vector if available.

−fea_E <on|off>
Add log of frame energy to the feature vector. Log energy is defined for every fea_kind as follows:

fea_kind = spec/logspec/lpa/lpc: log of frame energy after passing through Noise Reduction and Filter Bank blocks. It is identical to the log of the zeroth autocorrelation coefficient.

fea_kind = dctc: Energy is computed always before Filter Bank block. When Noise Reduction takes place after Filter bank, it is computed from input spectra; otherwise it is computed from spectra after Noise Reduction.

NOTE: These definitions can be overriden by the −fea_rawenergy option.

−fea_rawenergy <on|off>
Compute frame energy directly from the input signal frame before doing anything (preemphasis, Hamming, FFT), ignoring fea_kind.

−fea_lifter <*int*>
Cepstrum liftering constant. Value of 1 means no liftering. Otherwise it is defined as in HTK book.

−fea_Z_block <*int*>
Average cepstrum is computed on the basis of moving average over the fixed time interval. Integer value of fea_Z_block correspodns to the length of moving window in miliseconds across which

average cepstrum is computed. Value 5000 means the window length equal to 5s, value of -1 means no CMS.

−fea_Z_exp <*int*>
Average cepstrum is computed on the basis of recursive exponential moving average. Integer value of fea_Z_exp corresponds to the time constant in miliseconds (equivalent window size) across which average cepstrum is computed. Value 5000 means the window length equal to 5s, value of -1 means no CMS.

−stat_cmvn <*filename*>
The cepstrum mean and variance statistics are computed on the basis of given input list for feature extraction (usually grouping utterances of particular speakers; see examples/run.sh script) and saved to the specified output file for further application using .apply_cmvn option.

−apply_cmvn <*filename*>
The cepstrum mean and variance normalization is done on the basis of loaded CMVN statistics precomputed using the -stat_cmvn option and saved in the specified file.

PRESETS:

−preset <*type*>
Apply a preset to the above options. This macro option behaves just like any other option. Applied settings can be overriden by any command line option following the −preset option.

To see exactly what has changed after using the −preset option, it is recommended to use −v option and check the program output. For macro definitions, see below.

<*type*> = mfcc − MFCC computation similar to what HTK does

equivalent is: "-fb_scale mel -fb_shape triang -fb_power on -fb_definition 1-26/26filters -nr_mode none -fb_eqld off -fb_inld off -fea_kind dctc -fea_ncepcoefs 12 -fea_c0 on -fea_E off -fea_lifter 22 -fea_rawenergy off"

plpc − suitable for PLP computation similar to the original Hermansky paper (Bark scale, trapezoidal filters).

equivalent is: "-fb_scale bark -fb_shape trapez -fb_power on -fb_definition 1-15/15filters -nr_mode none -fb_eqld on -fb_inld on -fea_kind lpc -fea_lporder 12 -fea_ncepcoefs 12 -fea_c0 on -fea_E off -fea_lifter 22 -fea_rawenergy off"

exten − suitable for speech enhancement using Extended Spectral Subtraction.

equivalent is: "-w 32 -s 10 -fb_definition none -fb_scale none -fb_shape none -nr_a 2 -fb_eqld off -fb_inld off -fb_power off -fb_norm off -nr_mode exten -fea_kind none -fea_c0 off -fea_E off -fea_lifter 0 -fea_rawenergy off"

Note that by default CtuCopy acts as a feature extractor with pseudo−PLP features, mel scale, triangular filters.

MISCELLANEOUS:

−verbose, −v
Verbose mode. Prints generally more, also prints all warnings and at the beginning also prints overall program settings. Highly recommended for debugging, optimizations and fine−tuning.

−quiet Suppress console output. Only error messages related to a premature program termination are printed.

−info Prints overall program settings at the beginning.

−C <*filename*>
Specifies configuration file. Configuration file acts as a set of command line options. Configuration file is always parsed before any other options on the command line.

SYNTAX: It is a text file with one command line option per line. Empty lines and whitespace are allowed. Comments are allowed, they begin with character #. Any text following # up to the end of line is ignored.

−h, −−help
Print brief version of this text and exit.

EXAMPLES

Speech enhancement with one file:

ctucopy -preset exten -fs 16000 -format_in wave -format_out wave -i input.wav -o output.wav -v

This loads a preset for speech enhancement using extended spectral subtraction, then sets sampling frequency to 16 kHz, sets input and output formats to MS wave and after a specification of input and output files sets the verbose flag to on so that program prints out full settings and also number of frames processed.

Online speech enhancement:

ctucopy -preset exten -fs 8000 -format_in raw -format_out raw -online_on -online_out < *input* **>** *output*

Reads 16 bit mono PCM data from *stdin* until EOF in machine native byte order and writes the enhanced output to *stdout*.

Feature extraction using default settings:

ctucopy -fs 8000 -format_in wave -format_out htk -endian_out big -S list.txt

Reads MS Wave files specified in the first column of the file list.txt, computes 13 speech features per frame and saves the results to HTK files specified in the second column of the file list.txt.

Feature extraction of HTK MFCCs:

ctucopy -fs 8000 -format_in wave -format_out htk -endian_out big -preset mfcc -S list.txt

The same as above, but with feature extraction settings suitable for MFCC features.

Feature extraction of original PLPs to a pfile:

ctucopy -fs 8000 -format_in wave -format_out pfile=features.pfile -preset plp -S list.txt

Using config file with the settings from the previous example:

ctucopy -C config.txt

Contents of config.txt:

```
# CtuCopy config file
-fs 8000         # sampling freq
-format_in wave   # MS wave
-format_out pfile=features.pfile
-preset plp
-S list.txt       # 2nd column of list.txt will be ignored
```

Advanced example of feature extraction:

ctucopy -format_in raw -endian_in big -fs 8000 -format_out htk -endian_out big -S list.txt -preem 0.97 -fb_scale expolog -fb_shape rect -fb_eqld off -fb_definition 0-3000Hz:3-5/5filters,3000-4000Hz:1-1/1filters -nr_mode 2fwss -vad burg -nr_when beforeFB -fea_kind lpc -v

Reads raw 16 bit mono PCM files in big endian coding from the input and saves the output to HTK files. Uses preemphasis. Filter bank is designed on expolog scale with rectangular filters with no overlap. Filter bank consists of the following filters. First, take the expolog scale and use an area from 0 Hz to 3000 Hz. Add to the filter bank the 3rd, 4th and 5th filter out of 5 filters that would be placed equidistantly in that part of frequency axis. Second, add one more filter to the filter bank that starts at 3000 Hz and ends at 4000 Hz. For noise suppression use two-pass spectral subtraction algorithm with internal Burg's VAD and perform the noise suppression before the filter bank. Compute LP cepstral coefficients. Be verbose so that settings can be double checked.

The other examples, how to work with ctucopy4 (CMVN, CMS, output in KALDI format) tool you can find in examples/run.sh script.

BUGS

Please report all bugs not mentioned below to the author.

- In speech enhancement mode the amplitude spectrum is modified. It affects the dynamics of the signal which does not necessarily fit the 16 bit range upon reconstruction. Thus, when a signal sample is bigger than the available dynamic range and the -v option is set, a warning message is printed locating the problematic sample and the sample is clipped. When the sample is larger than (2 x max value), then a warning message is printed always (unless quiet mode) and the sample is clipped.
- In speech enhancement mode the input and output signal lengths do not generally match. The input signal is read frame by frame until there are not enough samples for a full frame. It means the signal is cut at the position of the last available frame.
- MS Wave input and output formats do not support switching of byte order. They are always read in machine native format.
- When an external VAD file is used, there is no guarantee that the framing matches the input signal. User has to check this.
- In online mode no headers are written at the output (including HTK format) and for MS Wave input the number of samples from the file header is ignored.
- RASTA filtering is not implemented in current version.
- Filter bank design: In case of rectangular window, filters are designed not to have any overlap. In any subset of filters (specified with start and stop frequencies) the boundary bins are by default included so that the design is intuitive. However, if there exist any two subsets that are joined (one ends at the point where the other begins), the bin common to both is only present in the lower subset so that there is no overlap

"in the middle".

AUTHORS

Petr Mizera <petr.mizera8@gmail.com>
Petr Fousek <p.fousek@gmail.com>
Petr Pollak <pollak@fel.cvut.cz>
with kind help of Prague SpeechLab members.

HISTORY

The first ctucopy version was built on *exten*, an original implementation of Extended Spectral Subtraction by Pavel Sovka, Petr Pollak and Jan Kybic,
P. Sovka, P. Pollak, J. Kybic, "Extended Spectral Subtraction", EUSIPCO'96, Trieste, 1996.

Once completed, it was published as an open source project on Interspeech 2003,
P. Fousek, P. Pollak, "Additive Noise and Channel Distortion-Robust Parametrization Tool - Performance Evaluation on Aurora 2 & 3", Eurospeech'03, Geneva, 2003.

Later within a study of Lombard effect, an implementation of filter banks was much generalized and rewritten,
H. Boril, P. Fousek, P. Pollak, "Data-Driven Design of Front-End Filter bank for Lombard Speech Recognition", ICSLP'06, Pittsburgh, 2006.

In 2012, as a reaction to enquiries of commercial subjects, ctucopy was released under Apache 2.0 licence.

Next work was focused on various cepstral normalization techniques like CMVN, various CMS.
M. Borsky, P. Mizera, P. Pollak, "Noise and Channel Normalized Cepstral Features for Far-Speech Recognition", In Speech and Computer. Cham: Springer International Publishing AG, 2013.

The support KALDI format was added in work on ISP 2015.
P. Mizera, P. Pollak, M. Borsky, A. Kolman, M. Ernestus, "Towards the DNN-HMM Recognition of Czech Casual Speech", SUBMITTED TO Interspeech 2015.

COPYRIGHT

VERSION

This document is valid for CtuCopy versions 3.0 - 4.0