openSMILE – The Munich Versatile and Fast Open-Source Audio Feature Extractor

Florian Eyben
Institute for Human-Machine Communication
Technische Universität München
80290 München, Germany
eyben@tum.de

Martin Wöllmer
Institute for Human-Machine Communication
Technische Universität München
80290 München, Germany
woellmer@tum.de

Björn Schuller
Institute for Human-Machine Communication
Technische Universität München
80290 München, Germany
schuller@tum.de

ABSTRACT

We introduce the openSMILE feature extraction toolkit, which unites feature extraction algorithms from the speech processing and the Music Information Retrieval communities. Audio low-level descriptors such as CHROMA and CENS features, loudness, Mel-frequency cepstral coefficients, perceptual linear predictive cepstral coefficients, linear predictive coefficients, line spectral frequencies, fundamental frequency, and formant frequencies are supported. Delta regression and various statistical functionals can be applied to the low-level descriptors. openSMILE is implemented in C++ with no third-party dependencies for the core functionality. It is fast, runs on Unix and Windows platforms, and has a modular, component based architecture which makes extensions via plug-ins easy. It supports on-line incremental processing for all implemented features as well as off-line and batch processing. Numeric compatibility with future versions is ensured by means of unit tests. openSMILE can be downloaded from http://opensmile.sourceforge.net/.

Categories and Subject Descriptors

H.5.5 [**Information Systems Applications**]: Sound and Music Computing

General Terms

Design, Performance

Keywords

audio feature extraction, statistical functionals, signal processing, music, speech, emotion

MM'10, October 25–29, 2010, Firenze, Italy.

1. INTRODUCTION

Feature extraction is an essential part of many audio analysis tasks, e.g. Automatic Speech Recognition (ASR), analysis of paralinguistics in speech, and Music Information Retrieval (MIR). There are a few freely available feature extraction utilities which, however, are mostly designed for a special domain, such as ASR or MIR (see section 2). Moreover, they are either targeted at off-line data processing or are libraries, which do not offer a ready-to-use, yet flexible, feature extractor. Tools for off-line feature extraction are useful for research tasks, but when it comes to building a live demonstrator system (e.g. the SEMAINE system[1]) or a commercial application where one wants to use the exact same features as used in research work, something different is needed.

We thus introduce openSMILE[2], a novel open-source feature extractor for incremental processing. *SMILE* is an acronym for *Speech and Music Interpretation by Large-space Extraction*. Its aim is to unite features from two worlds, speech processing and Music Information Retrieval, enabling researchers in either domain to benefit from features from the other domain. A strong focus is put on fully supporting real-time, incremental processing. openSMILE provides a simple, scriptable console application where modular feature extraction components can be freely configured and connected via a single configuration file. No feature has to be computed twice, since output from any feature extractor can be used as input to all other feature extractors internally. Unit tests are provided for developers to ensure exact numeric compatibility with future versions.

Even though openSMILE's primarily intended area of use is audio feature extraction, it is principally modality independent, e.g. physiological features such as heart rate, EEG, or EMG signals can be analysed with openSMILE using audio processing algorithms. An easy plugin interface moreover provides the ability to extend openSMILE with one's own components, thus virtually being able to solve any feature extraction task and thereby using existing components as building blocks.

In the following we give an overview on related tools in section 2, describe openSMILE's design principles in section 3 and the implemented features and descriptors in section 4. We provide computation time benchmarks in section 5 and summarise this overview paper in section 6.

[1] `http://www.semaine-project.eu/`
[2] Available at: `http://opensmile.sourceforge.net/`

2. RELATED TOOLKITS

Related feature extraction tools used for speech research include e.g. the *Hidden Markov Model Toolkit* (*HTK*) [15], the *PRAAT* Software [3], the *Speech Filing System*[3] (*SFS*), the Auditory Toolbox[4], a Matlab^{TM} toolbox[5] by Raul Fernandez [6], the Tracter framework [7], and the *SNACK*[6] package for the Tcl scripting language. However, not all of these tools are distributed under a permissive open-source license, e.g. *HTK* and *SFS*. The *SNACK* package is without support since 2004.

For Music Information Retrieval many feature extraction programs under a permissive open-source license exist, e.g. the lightweight ANSI C library *libXtract*[7], the Java based *jAudio* extractor [9], the Music Analysis, Retrieval and Synthesis Software *Marsyas*[8], the *FEAPI* framework [8], the MIRtoolbox[9], and the *CLAM* framework [1].

However, very few feature extraction utilities exist that unite features from both speech and music domains. While many features are common, MFCC or LPC are used primarily for speech, for example, and e.g. CHROMA features or algorithms for estimating multiple fundamental frequencies are mostly found in music applications. Next to low-level audio features, *functionals* mapping time series of variable length to static values are common e.g. for emotion recognition or music genre discrimination. Such mappings are also referred to as *aggregate features* or *feature summaries*. Further, delta coefficients, moving average, or various filter types are commonly applied to feature contours. Hierarchies of such post-processing steps have proven to lead to more robust features, e.g. in [13], hierarchical functionals, i.e. 'functionals of functionals' are used for robust speech emotion recognition.

For an application or demonstrator system which is a result of research work, it is convenient to use the same feature extraction code in the live system as used to produce published results. To achieve this goal, we require incremental processing of the input with a delay as small as possible.

3. OPENSMILE'S ARCHITECTURE

This section addresses the problems that were considered during planning of openSMILE's architecture and summarises the resulting architecture. A more detailed description can be found in the openSMILE documentation.

In order to address deficiencies in existing software packages, such as the lack of a comprehensive cross-domain feature set, the lack of flexibility and extensibility, and the lack of incremental processing support, the following requirements had to be – and were – met:

- **Incremental processing**, where data from an arbitrary input stream (file, sound card, etc.) is pushed through the processing chain sample by sample and *frame by frame* (see figure 1).
- **Ring-buffer memory** for features requiring temporal context and/or buffering, and for *reusability of data*, i.e. to avoid duplicate computation of data used by multiple feature extractors such as FFT spectra (see figure 1, right).
- **Fast and lightweight algorithms** carefully implemented in C/C++, no third-party dependencies for the core functionality.
- **Modular architecture** which allows for arbitrary feature combination and easy addition of new feature extractor components by the community via a well structured API and a run-time plug-in interface.
- **Configuration** of feature extractor parameters and component connections in a single configuration file.

Moreover, the extractor is easy to compile on many commonly used platforms, such as Windows, Unix, and Mac.

Figure 1 (left) shows the overall data-flow architecture of openSMILE, where the *Data Memory* is the central link between all *Data Sources* (components that write data from external sources to the data memory), *Data Processors* (components which read data from the data memory, modify it, and write it back to the data memory), and *Data Sinks* (components that read data from the data memory and write it to external places such as files).

The ring-buffer based incremental processing is illustrated in figure 1 (mid). Three levels are present in this setup: wave, frames, and pitch. A cWaveSource component writes samples to the 'wave' level. The write positions in the levels are indicated by the vertical arrows. A cFramer produces frames of size 3 from the wave samples (non-overlapping), and writes these frames to the 'frames' level. A cPitch (simplified for the purpose of illustration) component extracts pitch features from the frames and writes them to the 'pitch' level. Since all boxes in the plot contain values (=data), the buffers have been filled, and the write pointers have been warped. Figure 1 (right) shows the incremental processing for higher order features. Functionals (max and min) over two frames (overlapping) of the pitch features are extracted and saved to the level 'func'.

The size of the buffers must be adjusted to the size of the block a reader or writer reads/writes from/to the data memory at once. In the above example the read blocksize of the functionals component would be 2 because it reads 2 pitch frames at once. The input level buffer of 'pitch' must be at least 2 frames long, otherwise the functionals component will not be able to read a complete window from this level. openSMILE handles this adjustment of the buffersize automatically.

To speed up computation, openSMILE supports multithreading. Each component in openSMILE can be run in a separate thread. This enables parallelisation of the feature extraction process on multi-core machines and reduces computation time when processing large files.

4. AVAILABLE FEATURE EXTRACTORS

openSMILE is capable of extracting Low-Level Descriptors (LLD) and applying various filters, functionals, and transformations to these. The LLD currently implemented are listed in table 1.

[3] http://www.phon.ucl.ac.uk/resource/sfs/
[4] http://cobweb.ecn.purdue.edu/~malcolm/interval/1998-010/
[5] http://affect.media.mit.edu/publications.php
[6] http://www.speech.kth.se/snack/
[7] http://libxtract.sourceforge.net/
[8] http://marsyas.sness.net/
[9] https://www.jyu.fi/hum/laitokset/musiikki/en/research/coe/materials/mirtoolbox

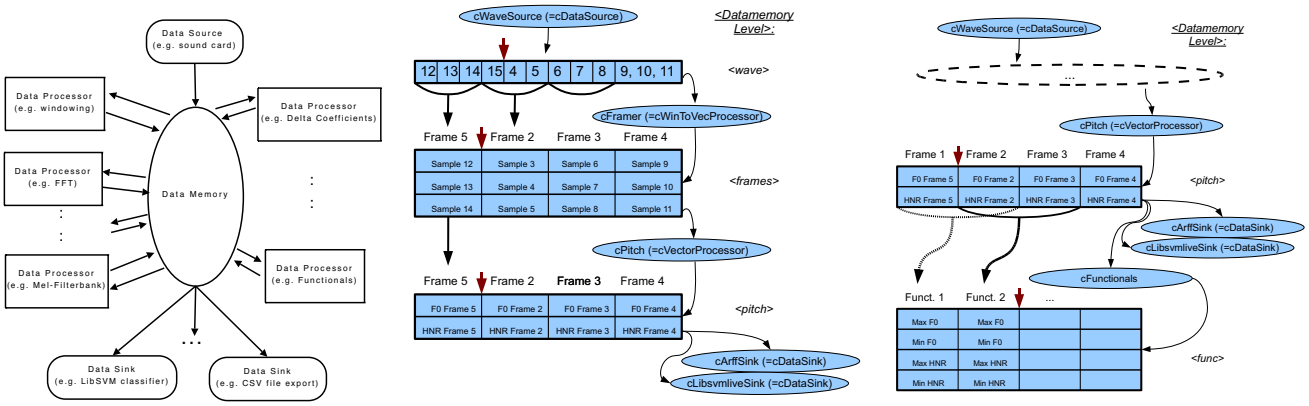

Figure 1: Sketch of openSMILE's architecture (left) and incremental data-flow in ring-buffer memories (centre and right); the (red) arrow (pointing in between the columns) indicates the current write pointer.

Feature Group	Description
Waveform	Zero-Crossings, Extremes, DC
Signal energy	Root Mean-Square & logarithmic
Loudness	Intensity & approx. loudness
FFT spectrum	Phase, magnitude (lin, dB, dBA)
ACF, Cepstrum	Autocorrelation and Cepstrum
Mel/Bark spectr.	Bands 0-N_{mel}
Semitone spectr.	FFT based and filter based
Cepstral	Cepstral features, e.g. MFCC, PLP-CC
Pitch	F_0 via ACF and SHS methods Probability of Voicing
Voice Quality	HNR, Jitter, Shimmer
LPC	LPC coeff., reflect. coeff., residual Line spectral pairs (LSP)
Auditory	Auditory spectra and PLP coeff.
Formants	Centre frequencies and bandwidths
Spectral	Energy in N user-defined bands, multiple roll-off points, centroid, entropy, flux, and rel. pos. of max./min.
Tonal	CHROMA, CENS, CHROMA-based features

Table 1: openSMILE's low-Level descriptors.

Category	Description
Extremes	Extreme values, positions, and ranges
Means	Arithmetic, quadratic, geometric
Moments	Std. dev., variance, kurtosis, skewness
Percentiles	Percentiles and percentile ranges
Regression	Linear and quad. approximation coefficients, regression err., and centroid
Peaks	Number of peaks, mean peak distance, mean peak amplitude
Segments	Number of segments based on delta thresholding, mean segment length
Sample values	Values of the contour at configurable relative positions
Times/durations	Up- and down-level times, rise/fall times, duration
Onsets	Number of onsets, relative position of first/last on-/offset
DCT	Coefficients of the Discrete Cosine Transformation (DCT)
Zero-Crossings	Zero-crossing rate, Mean-crossing rate

Table 2: Functionals (statistical, polynomial regression, and transformations) available in openSMILE.

The Mel-frequency features, Mel-Spectrum and Mel-Frequency Cepstral Coefficients (MFCC), as well as the Perceptual Linear Predictive Coefficients (PLP) can be computed exactly as described in [15], thus providing compatibility with the popular Hidden-Markov Toolkit (HTK).

Delta regression coefficients can be computed from the low-level descriptors, and a moving average filter can be applied to smooth the feature contours. All data vectors can be processed with elementary operations such as add, multiply, and power, which enables the user to create custom features by combining existing operations.

Next, functionals (statistical, polynomial regression coefficients, and transformations) listed in table 2 can be applied, e.g. to low-level features. This is a common technique, e.g. in emotion recognition ([11, 4]) and Music Information Retrieval [12]. The list of functionals is based on CEICES, where seven sites combined their features and established a feature coding standard that among others aims at a broad coverage of functional types [2]. Functionals can be applied multiple times in hierarchical structure as described in [13]).

Due to the modular architecture, it is possible to apply any implemented processing algorithm to any time series, i.e. the Mel-band filter-bank could be applied as a functional to any feature contour. This gives researchers an efficient and customisable tool to generate millions of novel features without adding a single line of C++ code.

To facilitate interoperability, feature data can be loaded from and saved to popular file formats such as WEKA ARFF [14], LibSVM format, Comma Separated Value (CSV) File, HTK [15] parameter files, and raw binary files (which can be read in, e.g. Matlab^{TM} or GNU Octave).

Live recording of audio and subsequent incremental extraction of features in real-time is also supported. A built-in voice activity detection can be used to pre-segment the

recorded audio stream in real-time, and on-line mean and variance normalisation as well as on-line histogram equalisation can be applied. Features extracted on-line can be directly visualised via gnuplot, which is a great feature especially for demonstration and teaching tasks.

5. PERFORMANCE

Since the main objective of openSMILE is real-time operation, run-time benchmarks for various feature sets are provided. Evaluation was done on a Ubuntu Linux machine with Kernel 2.6 and an AMD Phenom 64 bit CPU (only one core was used) at 2.2 GHz having 4 GB of DDR2" 800 RAM. All real-time factors (rtf) were computed by timing the CPU time required for extracting features from 10 minutes of monaural 16 kHz PCM (uncompressed) audio data.

Extraction of standard PLP and MFCC frame-based features with log-energy and 1st and 2nd order delta coefficients can be done with an rtf of 0.012. 250 k features (hierarchical functionals (2 levels) of 56 LLD (pitch, MFCC, LSP, etc.)) can be computed with an rtf of 0.044. Prosodic low-level features (pitch contour and loudness) can be extracted with an rtf of 0.026. This show the high efficiency of the code to compute functionals; most computation time is spent with tasks such as FFT or filtering during low-level descriptor extraction.

6. CONCLUSION AND OUTLOOK

We introduced openSMILE, an efficient, on-line (and also batch scriptable), open-source, cross platform, and extensible feature extractor implemented in C++. A well structured API and example components make integration of new feature extraction and I/O components easy. openSMILE is compatible with research tool-kits, such as HTK, WEKA, and LibSVM by supporting their data-formats. Although openSMILE is very new, it is already successfully used by researchers around the world. The openEAR project [5] builds on openSMILE features for doing emotion recognition. openSMILE was the official feature extractor for the INTERSPEECH 2009 Emotion Challenge [11] and the ongoing INTERPSEECH 2010 Paralinguistic Challenge. It has also been used for problems as exotic as classification of speaker height from voice characteristics [10].

Development of openSMILE is still active and even more features such as TEAGER energy, TOBI pitch descriptors, and psychoacoustic measures such as Sharpness and Roughness are considered for integration. Moreover, openSMILE will soon support MPEG-7 LLD XML output. In the near future we aim at linking to openCV[10], to be able to fuse visual and acoustic features. Due to openSMILE's modular architecture and the public source code, rapid addition of new and diverse features by the community is encouraged. Future work will focus on improved multithreading support and cooperation with related projects to ensure coverage of a broad variety of typically employed features in one piece of fast, lightweight, flexible open-source software.

Acknowledgment

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 211486 (SEMAINE).

[10] `http://opencv.willowgarage.com/`

7. REFERENCES

[1] X. Amatriain, P. Arumi, and D. Garcia. A framework for efficient and rapid development of cross-platform audio applications. *Multimedia Systems*, 14(1):15–32, June 2008.

[2] A. Batliner, S. Steidl, B. Schuller, D. Seppi, K. Laskowski, T. Vogt, L. Devillers, L. Vidrascu, N. Amir, L. Kessous, and V. Aharonson. Combining efforts for improving automatic classification of emotional user states. In T. Erjavec and J. Gros, editors, *Language Technologies, IS-LTC 2006*, pages 240–245. Informacijska Druzba, 2006.

[3] P. Boersma and D. Weenink. Praat: doing phonetics by computer (v. 4.3.14). http://www.praat.org/, 2005.

[4] F. Eyben, M. Wöllmer, A. Graves, B. Schuller, E. Douglas-Cowie, and R. Cowie. On-line emotion recognition in a 3-d activation-valence-time continuum using acoustic and linguistic cues. *Journal on Multimodal User Interfaces*, 3(1-2):7–19, Mar. 2010.

[5] F. Eyben, M. Wöllmer, and B. Schuller. openEAR - introducing the munich open-source emotion and affect recognition toolkit. In *Proc. of ACII 2009*, volume I, pages 576–581. IEEE, 2009.

[6] R. Fernandez. *A Computational Model for the Automatic Recognition of Affect in Speech*. PhD thesis, MIT Media Arts and Science, Feb. 2004.

[7] P. N. Garner, J. Dines, T. Hain, A. El Hannani, M. Karafiat, D. Korchagin, M. Lincoln, V. Wan, and L. Zhang. Real-time asr from meetings. In *Proc. of INTERSPEECH 2009, Brighton, UK*. ISCA, 2009.

[8] A. Lerch and G. Eisenberg. FEAPI: a low level feature extraction plug-in api. In *Proc. of the 8th International Conference on Digital Audio Effects (DAFx)*, Madrid, Spain, 2005.

[9] D. McEnnis, C. McKay, I. Fujinaga, and P. Depalle. jaudio: A feature extraction library. In *Proc. of ISMIR 2005*, pages 600–603, 2005.

[10] I. Mporas and T. Ganchev. Estimation of unknown speaker's height from speech. *International Journal of Speech Technology*, 12(4):149–160, dec 2009.

[11] B. Schuller, S. Steidl, and A. Batliner. The INTERSPEECH 2009 emotion challenge. In *Proc. Interspeech (2009)*, Brighton, UK, 2009. ISCA.

[12] B. Schuller, F. Wallhoff, D. Arsic, and G. Rigoll. Musical signal type discrimination based on large open feature sets. In *Proc. of the International Conference on Multimedia and Expo ICME 2006*. IEEE, 2006.

[13] B. Schuller, M. Wimmer, L. Mösenlechner, C. Kern, D. Arsic, and G. Rigoll. Brute-forcing hierarchical functionals for paralinguistics: A waste of feature space? In *Proc. of ICASSP 2008*, April 2008.

[14] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, 2nd edition edition, 2005.

[15] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland. *The HTK book (v3.4)*. Cambridge University Press, Cambridge, UK, December 2006.