An Evaluation of Audio Feature Extraction Toolboxes

Moffat, D; Ronan, D; Reiss, JD; 18th International Conference on Digital Audio Effects (DAFx-15)

AN EVALUATION OF AUDIO FEATURE EXTRACTION TOOLBOXES

David Moffat, David Ronan, Joshua D. Reiss

Center for Digital Music
Queen Mary University of London
Mile End Road
London, E1 4NS
{d.j.moffat, d.m.ronan, joshua.reiss}@qmul.ac.uk

ABSTRACT

Audio feature extraction underpins a massive proportion of audio processing, music information retrieval, audio effect design and audio synthesis. Design, analysis, synthesis and evaluation often rely on audio features, but there are a large and diverse range of feature extraction tools presented to the community. An evaluation of existing audio feature extraction libraries was undertaken. Ten libraries and toolboxes were evaluated with the Cranfield Model for evaluation of information retrieval systems, reviewing the coverage, effort, presentation and time lag of a system. Comparisons are undertaken of these tools and example use cases are presented as to when toolboxes are most suitable. This paper allows a software engineer or researcher to quickly and easily select a suitable audio feature extraction toolbox.

1. INTRODUCTION

Audio feature extraction is one of the cornerstones of current audio signal processing research and development.

Audio features are contextual information that can be extracted from an audio signal. Features can be broken down into groups, as presented in the Cuidado Project [1], which includes definitions of a range of features.

Audio features can be applied to a range of research fields including:

- Feature extraction linked to audio effects [2]
- Statistical synthesis [3]
- Feature-based synthesis [4]
- Evaluating synthesis techniques [5]
- Similarity measures [6]
- Data classification [7]
- Data mining [8]

Although these problems are somewhat dissimilar in nature, they lean heavily on a set of related audio features. Low level features are computed directly from the audio signal, often in a frame-by-frame basis' such as zero-crossing rate, spectral centroid or signal energy, and generally have little perceptual relevance in comparison to higher level features, like chord or key of musical piece, which hold greater semantic meaning. In MIR, it is common to refer to audio features or descriptors, whereas, in psychology distinctions are made between dimensions and features, where dimensions are continuous and features are a discrete. Descriptor is often used as a general term since it can refers to either continuous or discrete content.

Seventeen low level descriptors (LLDs) are defined in the MPEG-7 standard, with feature categorisation, for the purpose of performing audio similarity searching with metadata contained within an MPEG file [9, 10], and the Cuidado project takes this work further to define 54 audio features [1]. This project provides definitions of a range of features, grouping them and which are relevant as frame based features. These audio features can then processed with the aim of identifying some particular aspect of an audio signal. A good overview of features for extraction is presented in [11]

Consequently, a range of audio feature extraction libraries and toolboxes have been constructed. Some are built as workflow tools, with pre-processing and batch operations, some are written for algorithmic efficiency or parallelisation, some for specific programming environments or platforms. Despite significant growth and research in the field of audio signal processing and feature extraction, there has been little research on evaluating and identifying suitable feature extraction tools and their appropriate applications.

It has been identified that within music information retrieval (MIR) primarily focuses on precision and recall, which may be considered a limitation [12, 13]. Cleverdon et. al. developed a six point scale for measuring and evaluating information retrieval systems. This model is widely known as the Cranfield model of information retrieval evaluation [14]. The Cranfield model properties are: Coverage; Time Lag; Effort; Presentation; Precision; Recall. This model is an appropriate platform for evaluation and benchmarking of MIR systems.

This paper reviews and evaluates existing feature extraction libraries based on the Cranfield model. The properties of the model can be suitably related to the MIR feature extraction tool evaluation [15] and presents an evaluation based on the following criteria:

Coverage - The range of audio descriptor features presented by a toolkit, along with additional preprocessing or post processing functionality.

Effort - User Interface, how easily one can create a new specific query or modify queries, and appropriate documentation.

Presentation - File Output format options and consistency.

Time Lag - Computational Efficiency of each tool.

Precision and recall are both included as part of the Cranfield Model. However, within the case of evaluating feature extraction toolboxes, precision and recall are not considered applicable to the task, and as such are not used. Existing work discusses the merits of using precision and recall within MIR application [16].

This paper presents ten audio feature extraction toolboxes, evaluated based on the Cranfield model, as proposed in [12]. Section 3 compares the functionality of the tools with respect to the range

of audio features that can be extracted and any further pre or post processing that the tool implements. The interface options of each toolboxes is presented and discussed in Section 4. The output format of data of each toolbox is presented in Section 5 and the computational time is presented in Section 6.

2. EXISTING FEATURE EXTRACTION TOOLBOXES

There are a large number of audio feature extraction toolboxes available, delivered to the community in differing formats, but usually as at least one of the following formats:

- stand alone applications
- plug-ins for a host application
- software function library

To allow for delivery of tools, some APIs have been constructed to allow for feature extraction plug-ins to be developed. Vamp [17] is a C++ API specification which functions with the standalone applications such as Sonic Visualiser, a content and feature visualiser with Graphical User Interface (GUI) and its command line interface (CLI) counterpart Sonic Annotator [18]. The Vamp Plugin API is an independent plugin development framework and as a result plugin libraries have been developed by numerous research labs and academic institutions. However due to the nature of the framework, it is not possible to create plug-ins that depend on pre-existing plug-ins. This results in multiple implementations and instances of certain features being calculated, which causes potential system inefficiencies. Feature Extraction API (FEAPI) is another plugin framework API in C and C++ [19], though it less commonly used than the VAMP plugin format. There are also feature extraction libraries that provide their own plugin API for extending their stand alone system [20], though this is less common. There has been a rise in MIR web services, such as the web based audio feature extraction API produced by Echo Nest[1], where users submit files online and receive XML descriptions. These tools have resulted in large music feature datasets, such as the Million Song Dataset [21].

The feature extraction tools that are evaluated in this paper are:

Aubio A high level feature extraction library that extracts features such as onset detection, beat tracking, tempo, melody [22].

Essentia Full function workflow environment for high and low level features, facilitating audio input, preprocessing and statistical analysis of output. Written in C++, with Python binding and export data in YAML or JSON format. [23].

jAudio Java based stand alone application with Graphic User Interface (GUI) and CLI. Designed for batch processing to output in XML format or ARFF for loading into Weka [20].

Librosa API for feature extraction, for processing data in Python [24]

LibXtract Low level feature extraction tool written with the aim of efficient realtime feature extraction, originally in C but now ported to Max-MSP, Pure Data, Super Collider and Vamp formats [25].

Marsyas Full real time audio processing standalone framework for dataflow audio processing with GUI and CLI. This programme includes a low level feature extraction tool built in C++, with ability to perform machine learning and synthesis within the framework.The feature extraction aspects have also been translated to Vamp plugin format [26].

[1] http://developer.echonest.com/

Meyda Web Audio API based low level feature extraction tool, written in Javascript. Designed for web browser based efficient real time processing [27].

MIR Toolbox Audio processing API for offline extraction of high and low level audio features in Matlab. Includes preprocessing, classification and clustering functionality along with audio similarity and distance metrics as part of the toolbox functionality. Algorithms are fragmented allowing detailed control with simple syntax, but often suffers from standard Matlab memory management limitations [28].

Timbre Toolbox A Matlab toolbox for offline high and low level feature extraction. A toolbox that provides different set of features to the MIR Toolbox, specifically made efficient for identifying timbre and to fulfil the Cuidado standards [29].

YAAFE Low level feature extraction library designed for computational efficiency and batch processing by utilising data flow graphs, written in C++ with a CLI and bindings for Python and Matlab [30].

This list is not exhaustive, as there are many other feature extraction tools out there [31, 32, 33, 34]. However the list of tools was designed with popularity, programming environment range and how recently it has been updated all being taken into consideration.

3. COVERAGE

The coverage of an information retrieval system can be defined as the extent to which all relevant matters are covered by the system. Within the context of audio feature extraction tools, the coverage can be considered as the range of features a tool can extract. This section presents the features provided by each toolbox, relative to the total number of unique features from all presented tool boxes and the features from the MPEG-7 and Cuidado standard sets of audio descriptors. The relative importance of audio features is heavily context based. To provide a meaningful measure of the relative importance of audio features within each toolbox, the toolboxes will be compared to their compliance with the MPEG-7 and Cuidado standards. Additional functionality, including preprocessing and post processing available with each feature extraction tool will also be discussed. The accuracy of audio features or specific implementation detail is beyond the scope of this paper, but is discussed in [35].

The features available within each tool is evaluated, and a list of unique features is created. Each tool is then compared to the total list of unique features. Each tool is also evaluated based on the feature coverage when compared to the MPEG-7 and Cuidado standard feature sets. The results of this can be seen in Figure 1. It can be seen that Essentia provides the largest range of features, and is the only toolbox to produce 100% coverage of the MPEG-7 audio descriptors. Following this the MIR Toolbox and LibXtract both fulfill over 85% of the MPEG-7 and provide 85% and 75% of the features contained within the Cuidado project, respectfully. The Timbre Toolbox provides nearly 75% of the Cuidado feature set, however this may be unsurprising as they were both written by the same principal author. YAAFE, jAudio and Librosa provide fairly similar features sets, presenting between 30% and 38% of the MPEG-7 standard feature set, with YAAFE presenting some more perceptually motivated features than the others. Meyda and Aubio provide a relatively low number of features, however this is

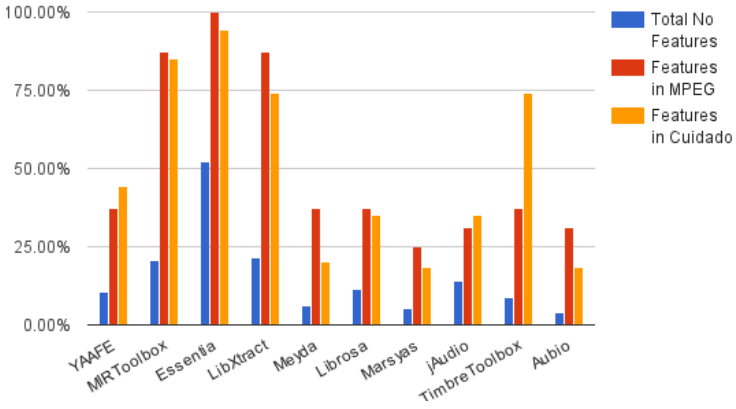

Figure 1: Graph of Percentage Coverage of Multiple Feature Sets

not without justification: Meyda is written for real time processing in the browser, and as such is inherently limited to a set of frame based features; Aubio is designed to focus on more high level feature extraction tools, and though providing a number of low level features, this is only to facilitate the high level feature extraction. Marsyas performs the worse in terms of feature range, complying to just 25% of MPEG-7 standard and 20% of the Cuidado standard. Marsyas is designed as an audio processing workflow, where clustering, classification and synthesis can all be performed within the entire workflow, so the range of available features may be limited, but the tool provides functionality beyond feature extraction. It is worth noting that only three features, spectral centroid, spectral rolloff and signal energy, are present in all the toolboxes and just 30 features are present in more than half of the toolboxes, and so attention must be paid if specific features are required.

Table 1 shows that LibXtract and Meyda do not provide any high level features. jAudio provides a limited range of high level features, such as beat histogram, and strongest beat. Aubio provides a limited range of low level features, such as spectral centroid, spread, skew and kurtosis. Aubio is designed specifically as a high level feature extraction tool, with particular focus on segmentation, whereas LibXtract, Meyda and jAudio are principally designed to extract low level features.

Additional functionality, such as preprocessing or post processing, is provided by a range of tools. Pre processing is an important aspect of evaluating any audio processing system, as it allows the user to be confident of a classification in any low quality environment where the audio may be degraded [36]. The resample function allows a standardisation of sample rates within a toolbox, which can be used to ensure that expected results for spectral comparisons are within the same range. For example, if a sample rate of 96kHz is used, it would be possible to have a spectral centroid of 30kHz, which has no perceptual meaning, compared to a file sampled at 44.1kHz, where a 30kHz spectral centroid would be impossible. As such standardisation of sample rates is an important factor that all feature extraction environments can provide. The quality of the resample method is an important attribute to consider [37], but is beyond the scope of this paper.

It can be seen from Table 1 that Aubio, Essentia, jAudio, Librosa, Marsyas and YAAFE all provide the user with some resample function as part of the toolbox, where as Meyda, MIR Toolbox and Timbre Toolbox all inherit a resample function from their native environments, as Web Audio API and Matlab both have resample functions built in. LibXtract does not provide a resample function, however, if used as a Vamp plugin, many Vamp hosts do contain resample functions.

Clustering, as a post processing tool, is also a useful functionality for many MIR processes. The post processing tools allow the user to directly analyse the output results as part of a single process. Essentia, Marsyas and MIR Toolbox all provide some from of clustering algorithm within them, and jAudio, Marsyas and MIR Toolbox can export files directly to ARFF format for loading directly into Weka, a data mining and clustering tool [38].

Essentia, MIR Toolbox and LibXtract produce a strong range of feature coverage, and the Timbre Toolbox covers the Cuidado feature set well. In terms of feature range these tools seem to perform better than many other existing tools. Essentia and MIR Toolbox both provide a powerful range of additional pre and post processing tools to benefit the user.

4. EFFORT

Effort is used to define how challenging a user finds a system to use, and whether any user experience considerations have been made while developing a system. Within this section, effort is evaluated relative to the user interface that is provided, whether it is a Graphical User Interface (GUI), Command Line Interface (CLI) or an Application Program Interface (API). The existence and quality of documentation and suitable examples is evaluated. The purpose is to identify how intuitively a tool's interface is presented to a user.

Table 1 outlines the user interfaces presented by each of the feature extraction tools. It can be seen that jAudio is the only tool that comes with its own GUI, though Aubio, LibXtract and Marsyas all have GUI capabilities through virtue of being Vamp plug-ins, when paired with a visualisation tool such as Sonic Visualiser. CLI's are more common, as Aubio, Marsyas, jAudio and YAAFE come with complete command line interfaces and Essentia comes with a series of precompiled C++ examples that can be run from the command line. However, this limits control functionality, as all the control is included in the software implementation. LibXtract can also be controlled via command line, through the use of its Vamp plugin format and a Vamp CLI tool such as Sonic Annotator. All tools come with APIs, which means Librosa, Meyda, MIR Toolbox and Timbre Toolbox are all only presented as software APIs, and as such all require software implementation before feature extraction is possible.

There are five different APIs written for both C and Python. Four APIs are available for Matlab, including the Essentia Matlab project[2]. Java has three APIs and only a single API for Javascript, Pure Data, Max-MSP, Supercollider, Lua and R are provided. Although Python and C are common programming languages, it is believed that Matlab is one of the most common frameworks used within MIR [39]. Environments such as web audio, in Javascript, Pure Data and Max-MSP are much less common in the MIR and audio research field, but are advantageous as they are real time audio environments where features are calculated in realtime and as such are excellent for prototyping.

Most toolboxes have clear documentation with examples, but there is limited documentation for LibXtract, Meyda and the Timbre toolbox. Though the documentation is not unclear, all the other tools provide a lot more information regarding basic access and software applications. Similarly, all toolboxes supply basic ex-

Table 1: Overview of Feature Extraction Tools

	Aubio	Essentia	jAudio	Librosa	LibXtract	Marsyas	Meyda	MIR	Timbre	YAAFE
High level Features	Y	Y	N‡	Y	N	Y	N	Y	Y	Y
Low Level Features	N*	Y	Y	Y	Y	Y	Y	Y	Y	Y
Resample	Y	Y	Y	Y	N	Y	Y†	Y†	Y†	Y
Filter	N	Y	N	N	N	N	Y†	Y†	Y†	N
Clustering	N	Y	N§	N	N	Y§	N	Y§	N	N
Similarity	N	N	N	N	N	N	N	Y	N	N
Real Time	Y				Y	Y	Y			
Vamp Plugin	Y	N	N	N	Y	Y	N	N	N	N
GUI	Y+	N	Y	N	Y+	Y+	N	N	N	N
CLI	Y	YE	Y	N	Y+	Y	N	N	N	Y
APIs	C/C++ Python R PD	C/C++ Python MatlabO PD/Max-MSP	Java	Python	C/C++ Supercollider PD/Max-MSP Java	C/C++ Python Java Lua	JS	Matlab	Matlab	Matlab Python C/C++
Output	Vamp	YAML JSON	XML ARFF	CSV	Vamp XML	Vamp CSV ARFF		TSV ARFF	TSV	CSV HDF5

* = Except MFCC and FFT Statistics,
‡ = Some Mid-high level features but very limited,
† = As part of environment, not toolbox,
+ = As result of being Vamp plugin,
§ = Can produce ARFF files, designed for being read directly into Weka.
E = CLI is produced through C 'Extractor' files, with some examples provided.
O = A project for calling Essentia from Matlab has been developed.

amples of implementation, however YAAFE, Essentia, Aubio and MIR Toolbox all have a strong range of examples that run straight away. Marsyas has clear documentation and a range of examples from which to draw inspiration, but required the user to learn a proprietary language for use as part of the system.

In conclusion, when looking for a stand alone tool which covers a user flexibility and usability with a user interface, then the Vamp plugin route is a useful one to take. This provides a simple intuitive interface and any number of specific features can be loaded in as required. If batch processing is required, then either the GUI from jAudio or CLI from YAAFE are intuitive, flexible and simple to use. If a user requires a programming API, then, depending on their environment, there are potentially a range of tools. C, C++ Python and Matlab APIs are provided by a range of tools, with often multiple being offered by each toolkit, as can be seen in Table 1.

5. PRESENTATION

An important aspect of any information retrieval system is how the resulting information is presented back to the user. Within this section, the output format of data is discussed and the relative merits of each approach outlined.

Document output format is one of the most significant barriers in fully integrated workflow solutions within MIR [40]. Output format is important, primarily as it impacts the ease and format of analysis someone can carry out on a dataset. Usually, a user using a software API, stores values in a relevant data structure within the given development language and as such, file output format becomes irrelevant in this case.

XML, YAML and JSON are all standard structured data formats, that allow the presentation of hierarchical structures of data. HDF5 is also a hierarchical data structure specifically designed for efficient storage and accuracy of contents - as such is well suited to big data tasks. CSV and TSV are table structures that allow users to view data in most spreadsheet applications, and ARFF is also a table structure with specific metadata about each column format and available options. ARFF is specifically designed for use with Weka, which is a powerful data mining tool.

CSV and TSV formats are considered to be suitable output formats if the resulting value can be considered as a table, however within the feature extraction, generally there is a much more complex data structure than simply two dimensions. Features carry varying levels of complexity, as some features are global for a signal where as some are based on windowed frames of a signal. Some features, such as MFCC's produce 13 numerical values per frame. As such it seems suitable that the data format used to output these results can represent these hierarchical feature formats. JSON and XML file formats are well supported by almost all programming languages, so there should not be any issues with processing the data results. CSV is also well supported within programming languages, but the lack of complexity or data structure can lead to potential ambiguities or errors with data transfer. The benefits of producing ARFF files, which can be loaded direct into Weka allows the user a great range of data mining opportunities and should not be underestimated.

Although it is clear that the file format is reliant on further applications, any feature extraction library should be able to present its data output in a data structure to suitable represent the hierarchical nature of the data it intends to represent. YAAFE, Essentia, jAudio and LibXtract all provide some from of suitable data structure, however only jAudio can also pass files direct into Weka [38].

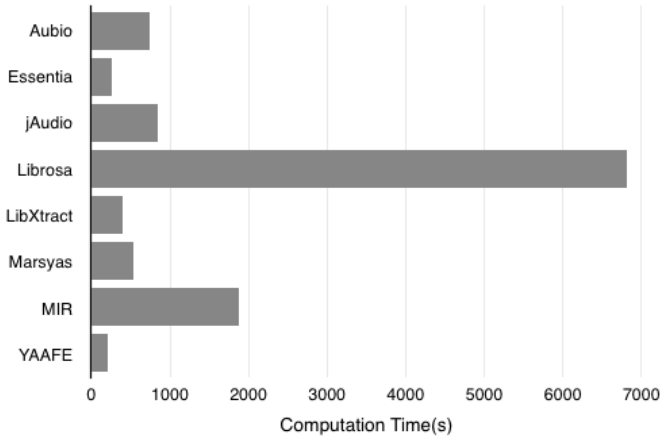

Figure 2: Graph of Computational Time of Feature Extraction Tools

Marsays and MIR Toolbox both allow for unstructured data, but can produce ARFF files for easy data mining, and Librosa and Timbre Toolbox will only allow users an unstructured data in a table format. YAAFE and jAudio are the only two applications that allow users the choice of structured or tabular data.

6. TIME LAG

Time lag is the measure of how long a given task will take to complete. Understanding the time necessary to perform a task, and comparing the relative speed of systems will give users an informed choice as to what system to use, particularly when they want to analyse large data sets. This section will discuss the computational complexity of the ten feature extractions tools and identify whether they are implemented in real time or if they are offline methods. There is existing work on implementing existing feature extraction tools in a distributed manner [41], but this is beyond the scope of this paper.

Meyda and the various LibXtract ports to Pure Data, Supercollider and Max-MSP are all designed to run in realtime. Each of these real time environments are provided with suitable feature extraction, which provides a user with powerful visualisation and real time interactivity but is less useful for users wishing to focus on offline approaches.

The offline approaches were all evaluated for computational efficiency. A dataset for evaluation is a subset of the Cambridge Multitrack Data Set[3]. 32 Different songs were used. The dataset consists of 561 tracks with an average duration of 106s is used, which totalled over 16.5hours of audio and 8.79Gb of data. Each toolbox is used to calculate the MFCC's from this data set, with a 512 sample window size and 256 sample hop size. The input audio is at a variety of different sample rates and bit depths to ensure that variable input file formats is allowable. This test is run on a MacBook Pro 2.9GHz i7 processor and 8Gb of RAM. The results are presented in Figure 2. The MFCCs were used, as they are a computational method, that exists within nine of the ten given tool boxes, and so should provide a good basis for comparison of computational efficiency. MFCCs are not computed by the Timbre Toolbox and Meyda will only run in real-time.

As can be seen from Figure 2, Yaafe is the fastest toolbox, processing over 16.5 hours of audio in just over 3 minutes 30s, with Essentia coming in as a close second place at 4 minutes 12s. LibXtract and Marsyas both completed in under 10 minutes, and both Aubio and jAudio ran in under 15 minutes. The MIR toolbox took over 31 minutes to run and Librosa took 1hour 53 minutes. It is evident that tools written in C or C++ run faster than tools written in Python or java.

[3]http://www.cambridge-mt.com/ms-mtk.htm

7. CONCLUSION

Ten audio feature extraction toolboxes are discussed and evaluated relative to four of the six criteria of the Cranfield Model.

Meyda and LibXtract provide excellent real time feature extraction tools in various programming environments. When high level features and segmentation is required, Aubio provides a simple and intuitive tool. It also provides a Vamp plugin format. When visualisation is required, for annotation or basic exploration of feature, using the Vamp plugin format is very powerful, and the combination of LibXtract and Marsysas as Vamp plug-ins provide excellent coverage of audio features. Research based in MATLAB should use the MIR Toolbox combined with the Timbre Toolbox for maximum feature coverage, or Essentia where computational efficiency is important with little sacrifice of feature range. Essentia performs the best with regards to computation, feature coverage and output presentation, with a range of APIs.

As the suitable feature extraction toolbox is entirely application dependent, there is no single case where a certain tool is better or more powerful than another. However suggestions for suitable toolboxes to use can be identified from Figure 3. When working on real time applications, either Meyda or LibXtract will be most suitable for applications, where as when working in an offline fashion, there is an option for either user interfaces or APIs. If a user interface is required then Vamp plug-ins, of LibXtract and Marsyas, are very powerful and advantageous tools, that can be hosted in either graphic interfaces or command line interfaces. jAudio provides a strong user interface with batch processing tool, but its range of features is limited. Essentia provides a strong CLI with large range of features, but low level of control, so implementation is required for accurate control of the features. If an API is required, then a range of example suggestions are proposed for some commonly used programming languages. A Java API is provided by jAudio, which is powerful and efficient, but performs on a reduced feature set. Strong Matlab APIs are provided by either a combination of MIR Toolbox and Timbre Toolbox or Essentia, with the 'Running essentia in matlab'.

8. ACKNOWLEDGMENTS

Funding for this research was provided by the Engineering and Physical Sciences Research Council (EPSRC).

References

[1] G. Peeters, "A large set of audio features for sound description (similarity and classification) in the cuidado project," Tech. Rep., IRCAM, 2004.

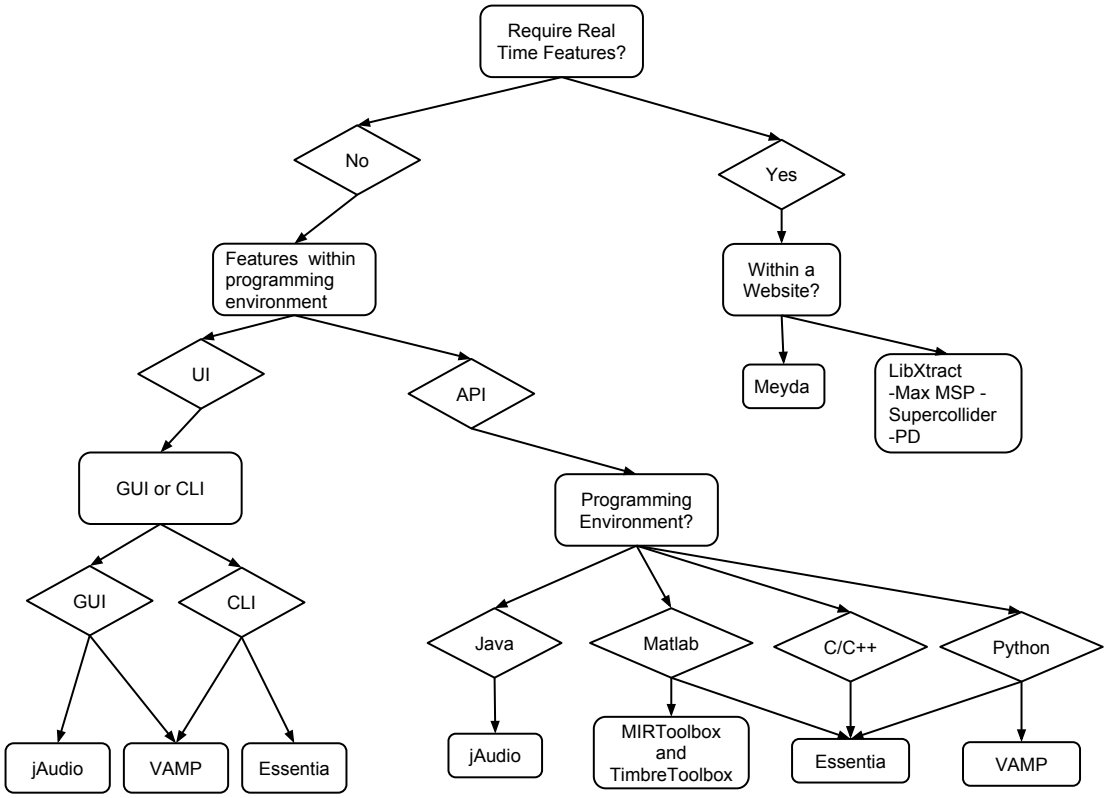

Figure 3: Flowchart to recommend what tool to use
N.B. Vamp = LibXtract and Marsyas Vamp Packages

[2] R. Stables, S. Enderby, B. De Man, G. Fazekas, and J. D. Reiss, "SAFE: A system for the extraction and retrieval of semantic audio descriptors," in *15th International Society for Music Information Retrieval Conference (ISMIR 2014)*, October 2014.

[3] J. H. McDermott and E. P. Simoncelli, "Sound texture perception via statistics of the auditory periphery: evidence from sound synthesis," *Neuron*, vol. 71, no. 5, pp. 926–940, 2011.

[4] M. D. Hoffman and P. R. Cook, "Feature-based synthesis: A tool for evaluating, designing, and interacting with music ir systems.," in *ISMIR*, 2006, pp. 361–362.

[5] S. Hendry and J. D. Reiss, "Physical modeling and synthesis of motor noise for replication of a sound effects library," in *Audio Engineering Society Convention 129*. Audio Engineering Society, 2010.

[6] B. Gygi, G. R. Kidd, and C. S. Watson, "Similarity and categorization of environmental sounds," *Perception & psychophysics*, vol. 69, no. 6, pp. 839–855, 2007.

[7] M. F. McKinney and J. Breebaart, "Features for audio and music classification.," in *ISMIR*, 2003, vol. 3, pp. 151–158.

[8] T. Li, M. Ogihara, and G. Tzanetakis, *Music data mining*, CRC Press, 2011.

[9] B. S. Manjunath, P. Salembier, and T. Sikora, *Introduction to MPEG-7: multimedia content description interface*, vol. 1, John Wiley & Sons, 2002.

[10] A. T. Lindsay and J. Herre, "MPEG-7 and MPEG-7 audio – an overview," *Journal of the Audio Engineering Society*, vol. 49, no. 7/8, pp. 589–594, 2001.

[11] D. Mitrović, M. Zeppelzauer, and C. Breiteneder, "Features for content-based audio retrieval," *Advances in computers*, vol. 78, pp. 71–150, 2010.

[12] J. D. Reiss and M. Sandler, "Beyond recall and precision: A full framework for MIR system evaluation," in *3rd Annual International Symposium on Music Information Retrieval, Paris, France*, 2002.

[13] J. D. Reiss and M. Sandler, "MIR benchmarking: Lessons learned from the multimedia community," *The MIR/MDL Evaluation Project White Paper Collection*, vol. 3, pp. 114–120, 2003.

[14] C. W. Cleverdon and M. Keen, "Aslib cranfield research project-factors determining the performance of indexing systems; volume 2, test results," Tech. Rep., Cranfield University, 1966.

[15] J. D. Reiss and M. Sandler, "Benchmarking music information retrieval systems," *JCDL Workshop on the Creation of Standardized Test Collections, Tasks, and Metrics for Music Information Retreival (MIR) and Music Digital Library (MDL) Evaluation*, pp. 37–42, July 2002.

[16] J. S. Downie, "The scientific evaluation of music information retrieval systems: Foundations and future," *Computer Music Journal*, vol. 28, no. 2, pp. 12–23, 2004.

[17] C. Cannam, "The vamp audio analysis plugin api: A programmer's guide," *Availble online: http://vamp-plugins. org/guide. pdf*, 2009.

[18] C. Cannam, C. Landone, and M. Sandler, "Sonic visualiser: An open source application for viewing, analysing, and annotating music audio files," in *Proceedings of the ACM Multimedia 2010 International Conference*, Firenze, Italy, October 2010, pp. 1467–1468.

[19] A. Lerch, G. Eisenberg, and K. Tanghe, "FEAPI: A low level feature extraction plugin api," in *Proceedings of the International Conference on Digital Audio Effects*, 2005.

[20] C. McKay, I. Fujinaga, and P. Depalle, "jAudio: A feature extraction library," in *Proceedings of the International Conference on Music Information Retrieval*, 2005, pp. 600–3.

[21] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *ISMIR 2011: Proceedings of the 12th International Society for Music Information Retrieval Conference, October 24-28, 2011, Miami, Florida*. University of Miami, 2011, pp. 591–596.

[22] P. M. Brossier, "The aubio library at MIREX 2006," *MIREX 2006*, p. 1, 2006.

[23] D. Bogdanov, N. Wack, E. Gómez, S. Gulati, P. Herrera, O. Mayor, G. Roma, J. Salamon, J. R. Zapata, and X. Serra, "Essentia: An audio analysis library for music information retrieval.," in *ISMIR*, 2013, pp. 493–498.

[24] B. McFee, M. McVicar, C. Raffel, D. Liang, and D. Repetto, "librosa: v0.3.1," Nov. 2014.

[25] J. Bullock and U. Conservatoire, "Libxtract: A lightweight library for audio feature extraction," in *Proceedings of the International Computer Music Conference*, 2007, vol. 43.

[26] G. Tzanetakis and P. Cook, "Marsyas: A framework for audio analysis," *Organised sound*, vol. 4, no. 03, pp. 169–175, 2000.

[27] H. Rawlinson, N. Segal, and J. Fiala, "Meyda: an audio feature extraction library for the web audio api," in *Web Audio Conference*. Web Audio Conference, 2015.

[28] O. Lartillot and P. Toiviainen, "A matlab toolbox for musical feature extraction from audio," in *International Conference on Digital Audio Effects*, 2007, pp. 237–244.

[29] G. Peeters, B. L. Giordano, P. Susini, N. Misdariis, and S. McAdams, "The timbre toolbox: Extracting audio descriptors from musical signals," *The Journal of the Acoustical Society of America*, vol. 130, no. 5, pp. 2902–2916, 2011.

[30] B. Mathieu, S. Essid, T. Fillon, J. Prado, and G. Richard, "YAAFE, an easy to use and efficient audio feature extraction software.," in *ISMIR*, 2010, pp. 441–446.

[31] W. Brent, *A timbre analysis and classification toolkit for pure data*, Ann Arbor, MI: MPublishing, University of Michigan Library, 2010.

[32] F. Eyben, F. Weninger, F. Groß, and B. Schuller, "Recent developments in openSMILE, the munich open-source multimedia feature extractor," in *Proceedings of the 21st ACM international conference on Multimedia*. ACM, 2013, pp. 835–838.

[33] D. L. Bryant, "Scalable audio feature extraction," M.S. thesis, University of Colorado Colorado Springs, 2014.

[34] F. Deliege, B. Y. Chua, and T. B. Pedersen, "High-level audio features: Distributed extraction and similarity search," in *Ninth International Conference on Music Information Retrieval*, 2008, pp. 565–570.

[35] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, and D. P. Ellis, "mir_eval: A Transparent Implementation of Common MIR Metrics," in *Proc. of the 15th International Society for Music Information Retrieval Conference, Taipei, Taiwan*, 2014.

[36] M. Mauch and S. Ewert, "The audio degradation toolbox and its application to robustness evaluation.," in *ISMIR*, 2013, pp. 83–88.

[37] A. Franck, "Performance evaluation of algorithms for arbitrary sample rate conversion," in *Audio Engineering Society Convention 131*, Oct 2011.

[38] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[39] K. R. Page, B. Fields, D. De Roure, T. Crawford, and J. S. Downie, "Reuse, remix, repeat: the workflows of MIR.," in *ISMIR*, 2012, pp. 409–414.

[40] M. A. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney, "Content-based music information retrieval: Current directions and future challenges," *Proceedings of the IEEE*, vol. 96, no. 4, pp. 668–696, 2008.

[41] S. Bray and G. Tzanetakis, "Distributed audio feature extraction for music.," in *ISMIR*, 2005, pp. 434–437.