MIR_EVAL

First author
Affiliation1
author1@ismir.edu

Second author
Retain these fake authors in submission to preserve the formatting

Third author
Affiliation3
author3@ismir.edu

ABSTRACT

Central to the field of MIR research is the evaluation of algorithms used to extract information from music data. We present mir_eval, an open source software library which provides a transparent and easy-to-use implementation of the most common metrics used to measure the performance of MIR algorithms. In the present work, we enumerate the metrics implemented by mir_eval and quantitatively compare each to an existing implementation. When the score reported by mir_eval differs substantially from the reference, we detail the differences in implementation. We also provide a brief overview of mir_eval's architecture, design, and intended use.

1. EVALUATING MIR ALGORITHMS

2. TASKS INCLUDED IN MIR_EVAL

2.1 Beat Detection

The aim of a beat detection algorithm is to report the times at which a typical human listener might tap their foot to a piece of music. As a result, most metrics for evaluating the performance of beat tracking systems involve computing the error between the estimated beat times and some reference list of beat locations. Many metrics additionally compare the beat sequences at different metric levels in order to deal with the ambiguity of tempo [4].

mir_eval includes the following metrics for beat tracking, which are defined in detail in [2]: The f-measure of the beat sequence, where an estimated beat is considered correct if it is sufficiently close to a reference beat; Cemgil's score, which computes the sum of Gaussian errors for each beat; Goto's score, a binary score which is 1 when some specific heuristic criteria are met, McKinney's P-score, which computes the cross-correlation of the estimated and reference beat sequences represented as impulse trains; continuity-based scores which compute the proportion of the beat sequence which is continuously correct; and finally the information gain of the beat error histogram to a uniform distribution.

2.2 Chord Recognition

Despite being one of the oldest MIREX tasks, evaluation methodology and metrics for automatic chord recognition is an ongoing topic of discussion. Several recent articles address issues and concerns with vocabularies, comparison semantics, and other lexicographical challenges unique to chord recognition []. Ultimately, the source of this difficulty stems from the inherent subjectivity in "spelling" a chord name and the level of detail a human observer can provide in a reference annotation [**?**]. As a result, a consensus has yet to be reached regarding the single best approach to comparing two sequences of chord labels, and instead are often compared over a set of rules, e.g Major-Minor, Sevenths, with or without inversions, and so on.

Thanks to the previous efforts of Harte [], text representations of chord labels adhere to a standardized format, consisting of a root, quality, extensions, and a bass note; of these, only the root is strictly required. However, in order to efficiently compare chords in a variety of different ways, it is helpful to first translate a given chord label $\mathcal{C}$ into a numerical representation, shown in Figure **??**. In this example, a $G:7(9)/5$ is mapped to split into 4 pieces of information: one, the root is mapped to an absolute pitch class $\mathcal{R}$, in $[0, 11]$, where $C \rightarrow 0$, $C\sharp/D\flat \rightarrow 1$, etc; two, the quality is mapped to a root-invariant 12-dimensional bit vector $\mathcal{Q}$ by setting the scale degrees of the quality; three, any extensions are applied (via addition or omission) to the quality bit vector as scale degrees in a single octave, resulting in pitch vector $\mathcal{P}$; and four, the bass interval (5) is translated to the relative scale degree in semitones $\mathcal{B}$. Note that the add-9 is rolled into a single octave as an add-2. This is a matter of convenience as extended chords (9's, 11's or 13's) are traditionally resolved to a single-octave equivalent, but the bit-vector representation could be easily expanded to represent such information.

Having gone through this bit of effort, it is now straightforward to compare chords along the five rules used in MIREX 2013:

1. Root:
   (a) $\mathcal{R}_{est} == \mathcal{R}_{ref}$
   (b) $\forall \mathcal{Q}_{ref}$
2. Major-Minor: Rule 1.a, plus
   (a) $\mathcal{Q}_{est} == \mathcal{Q}_{ref}$
   (b) $\mathcal{Q}_{ref} \in \{Maj, min\}$

3. Major-Minor w/Inversions: Rule 2, plus
    (a) $\mathcal{B}_{ref} \in \mathcal{Q}_{ref}$
4. Sevenths: Rule 1.1, plus
    (a) $\mathcal{Q}_{est} == \mathcal{Q}_{ref}$
    (b) $\mathcal{Q}_{ref} \in \{Maj, min, Maj7, min7, 7\}$
5. Sevenths w/Inversions: Rule 4, plus
    (a) $\mathcal{B}_{ref} \in \mathcal{Q}_{ref}$

Following recent trends in MIREX, an overall score is computed by weighting each comparison by the duration of its interval, over all intervals; stated another way, this is the piecewise continuous-time integral of the intersection of two chord sequences, $(\mathbf{C}_{ref}, \mathbf{C}_{est})$, expressed as follows:

$$S(\mathbf{C}_{ref}, \mathbf{C}_{est}) = \frac{1}{T} \int_{t=0}^{T} \mathcal{C}_{ref}(t) == \mathcal{C}_{est}(t) \quad (1)$$

Here, this is achieved here by forming the union of the boundaries in each sequence, and summing the time intervals of the correct ranges. Note that equivalence is subject to one of the rules defined previously.

Finally, the total score over a set of N items is given by a discrete summation, where the importance of each score, S_n, is weighted by the duration, T_n, of each annotation:

$$S_{total} = \frac{\sum_{n=0}^{N} T_n * S_n}{\sum_{n=0}^{N} T_n} \quad (2)$$

2.3 Pattern Discovery

This task aims to evaluate algorithms that identify musical patterns (i.e. short fragments or melodic ideas that repeat at least twice) both from audio and symbolic representations. Given the novelty of this task [1], the evaluation metrics are likely to be modified in further editions. Nonetheless, Collins put together all the previously existent metrics and a few new ones for this task in its first appearance in MIREX [1], which resulted in 19 different scores, each one (except for the two that evaluate the algorithm execution time) implemented in MIR-eval and described below:

- **Standard F-measure, Precision, and Recall ($\mathbf{F}_1$, P, R)**: This metric, composed of three scores, checks if the prototype patterns of the reference match possible key-transposed patterns in the prototype patterns of the estimations. Since the sizes of these prototypes must be equal, this metric is quite restrictive and it tends to be 0 most of the time (see 2013 MIREX results).
- **Establishment F-measure, Precision, and Recall ($\mathbf{F1}_{est}$, $\mathbf{P}_{est}$, $\mathbf{R}_{est}$)**: These scores evaluate the amount of patterns that were successfully identified by the estimated results, no matter how many occurrences they found. In other words, this metric captures the ability of the algorithm to *establish* that the estimated patterns are actually contained in the reference annotation.
- **Occurrence F-measure, Precision, and Recall ($\mathbf{F1}_{occ}$, $\mathbf{P}_{occ}$, $\mathbf{R}_{occ}$)**: Independently of how many patterns were correctly established, we may also want to know how well the algorithm finds all their respective occurrences throughout the piece. This metric aims to quantize this *occurrence* retrieval quality of the algorithm. This metric has a tolerance parameter c to allow differences when evaluating the similarity between occurrences, and in MIREX they use $c = .75$ and $c = .5$.
- **Three-layer F-measure, Precision, and Recall ($\mathbf{TLF}_1$, $\mathbf{P}_3$, $\mathbf{R}_3$)**: These scores can be seen as an improved version of the standard metrics. They capture both the establishment of the patterns and the occurrence retrieval in a single set of scores, being more permissive than the standard evaluation.
- **First N patterns metrics ($\mathbf{FFTP}_{est}$, FFP)**: This includes the first N patterns target proportion establishment recall, and the first N patterns three-layer precision. By analyzing the first N patterns only, we evaluate the ability of the algorithm of sorting the identified patterns based on their relevance. In MIREX, $N = 5$.

[1] The first year to appear on MIREX was 2013.

2.4 Segmentation

Evaluation criteria for segmentation fall into two categories: boundary annotation, and structural annotation. *Boundary annotation* is the task of predicting the times at which structural changes occur, such as when a *verse* transitions to a *refrain*. *Structural annotation* is the task of assigning labels to detected segments. The estimated labels may be arbitrary strings — such as A, B, C, *etc.* — and they need not describe functional concepts.

In both tasks, we assume that annotations express a partitioning of the track into intervals $(s_i, t_i)_{i=1}^{m}$, where $s_0 = 0$ denotes the beginning of track, t_m denotes the end of the track, and $t_i = s_{i+1}$. Structural annotation additionally requires that each interval be assigned a label y_i.

2.4.1 Boundary annotation

Within boundary annotation, there are two categories of evaluation metrics: detection, and deviation [9].

Boundary detection measures the precision, recall, and F-measure of boundary prediction within a window. Let B^R (B^E) denote the set of unique interval boundaries in the reference (estimated) annotation, and let W denote a window, typically either 0.5s or 3.0s. A *hit* is defined as a pair $b_e \in B^E$, $b_r \in B^R$ such that $|b_e - b_r| \leq W$.

No estimated boundary is counted as a hit for more than one reference boundary, and vice versa. This is accomplished by computing a maximum bipartite matching H_W between B^E and B^R (subject to the window constraint W)

using the Hopcroft-Karp algorithm [3]. This deviates from the greedy heuristic used in the MIREX implementation, but it is guaranteed to produce a correct matching. Precision and recall at window W are defined as

$$P_W(B^R, B^E) := \frac{|H_W|}{|B^E|} \tag{3}$$

$$R_W(B^R, B^E) := \frac{|H_W|}{|B^R|}, \tag{4}$$

and the corresponding F-measure is defined by taking their harmonic mean.

Boundary deviation is comprised of two scores, which measure the median time between a reference boundary and its nearest corresponding estimated boundary, and vice versa. These two metrics have previously been defined as *true-to-predicted* (T-to-P) and *predicted-to-true* (P-to-T), though in the terminology of this document, we use alternative naming of *reference-to-estimated* (R-to-E) and *estimated-to-reference* (E-to-R):

$$\text{R-to-E}(B^R, B^E) := \operatorname*{median}_{b_r \in B^R} \min_{b_e \in B^E} |b_r - b_e| \tag{5}$$

$$\text{E-to-R}(B^R, B^E) := \operatorname*{median}_{b_e \in B^E} \min_{b_r \in B^R} |b_r - b_e| . \tag{6}$$

2.4.2 *Structural annotation*

Two standard methods of evaluating structural annotation accuracy are *pairwise classification* [5] and conditional entropy [6]. In both methods, a collection of samples are generated by sampling the labels at time steps between 0 and the track duration. The ith sample is assigned a reference label y_i^R and and estimated label y_i^E. Our implementation follows the MIREX guidelines, and generates samples at a default rate of 10Hz.

Reference and estimated annotations must span identical time durations for annotation metrics to be well-defined. This is accomplished in our implementation by trimming or padding the estimated annotation to exactly match the start and end-times reported in the reference annotation, and synthesizing unique labels if necessary.

Given the labels for the samples $\{(y_i^R, y_i^E)\}_{i=1}^n$, the *pairwise classification* metrics are defined as precision, recall, and F-measure of label agreement over all unique, distinct pairs $i \neq j$. Let $A_R = \{\{i,j\} |\ y_i^R = y_j^R\}$ denote the set of similarly labeled frames in the reference, with A_E defined analogously for the estimation. Precision and recall are defined as

$$P_{\text{pair}} := \frac{|A_E \cap A_R|}{|A_E|} \tag{7}$$

$$R_{\text{pair}} := \frac{|A_E \cap A_R|}{|A_R|}. \tag{8}$$

Normalized conditional entropy scores are computed by estimating the conditional entropy of the estimated label y^E given the reference label y^R and vice versa. Let P denote the empirical joint distribution over reference and estimated labels:

$$P_{ij} \propto \left|\{t |\ y_t^R = i \wedge y_t^E = j\}\right| . \tag{9}$$

Let Y^R and Y^E denote random variables corresponding to the reference and estimated label for an arbitrary sample. The conditional entropies $H\left(Y^E |\ Y^R\right)$ and $H\left(Y^R |\ Y^E\right)$ are estimated from the empirical marginals $p_i^R = \sum_j P_{ij}$ and $p_j^E = \sum_i P_{ij}$ as described by Lukashevich [6].

The final scores are defined as *over-segmentation* (S_O) and *under-segmentation* (S_U):

$$S_\text{O} := 1 - \frac{H(E |\ R)}{\log_2 |\ell^E|} \tag{10}$$

$$S_\text{U} := 1 - \frac{H(R |\ E)}{\log_2 |\ell^R|}, \tag{11}$$

where ℓ^R and ℓ^E denote the sets of unique label values given in the reference and estimation.

2.5 Melody Extraction

2.5.1 *Task definition*

Melody extraction algorithms aim to produce a sequence of frequency values corresponding to the pitch of the dominant melody from a musical recording [8]. The estimated pitch is represented as a time series of fundamental frequency (f_0) values in Hz sampled on a fixed time grid (e.g. every 10 ms). To evaluate the estimated sequence, a reference sequence is generated by running a pitch tracker on the monophonic melody track (requiring access to the multi-track recording session) and manually correcting any mistakes made by the pitch tracker. The estimate is then evaluated against the reference by comparing the two frequency sequences on a frame-by-frame basis, and computing five global measures, first used in the MIREX 2005 AME evaluation [7]. The goal of these measures, defined below, is to assess the algorithm's performance on two subtasks of melody extraction: (1) pitch estimation, i.e. how well the algorithm estimates the pitch of the melody, and (2) voicing detection, i.e. how well the algorithm determines when the melody is present in a frame (a *voiced* frame) or absent (an *unvoiced* frame). To allow evaluation of these two subtasks independently, a melody extraction algorithm can provide a frequency estimate even for frames it has determined to be unvoiced.

2.5.2 *Evaluation measures*

The following definitions are taken from [8] with permission from the authors. Let the estimated melody pitch frequency vector be $\mathbf{f}$, and the true sequence (reference) be $\mathbf{f}^*$. Let us also define a voicing indicator vector $\mathbf{v}$, whose τ^{th} element $v_\tau = 1$ when a melody pitch is detected, with corresponding ground truth $\mathbf{v}^*$. We also define an "unvoicing" indicator $\bar{v}_\tau = 1 - v_\tau$. Recall that an algorithm may report an estimated melody pitch ($f_\tau > 0$) even for times where it reports no voicing ($v_\tau = 0$). Then the measures are:

- **Voicing Recall Rate**: The proportion of frames labeled as melody frames in the reference that are estimated as melody frames by the algorithm.

$$\text{Rec}_{\text{vx}} = \frac{\sum_\tau v_\tau v_\tau^*}{\sum_\tau v_\tau^*} \tag{12}$$

- **Voicing False Alarm Rate**: The proportion of frames labeled as non-melody in the reference that are mistakenly estimated as melody frames by the algorithm.

$$\text{FA}_{\text{vx}} = \frac{\sum_\tau v_\tau \bar{v}^*_\tau}{\sum_\tau \bar{v}^*_\tau} \tag{13}$$

- **Raw Pitch Accuracy**: The proportion of melody frames in the reference for which f_τ is considered correct (i.e. within half a semitone of the ground truth f^*_τ).

$$\text{Acc}_{\text{pitch}} = \frac{\sum_\tau v^*_\tau \mathcal{T}\left[\mathcal{M}(f_\tau) - \mathcal{M}(f^*_\tau)\right]}{\sum_\tau v^*_\tau} \tag{14}$$

where $\mathcal{T}$ is a threshold function defined by:

$$\mathcal{T}[a] = \begin{cases} 1 & \text{if } |a| < 0.5 \\ 0 & \text{if } |a| \geq 0.5 \end{cases} \tag{15}$$

and $\mathcal{M}$ maps a frequency in Hertz to a melodic axis as a real-valued number of semitones above an arbitrary reference frequency f_{ref}:

$$\mathcal{M}(f) = 12 \log_2 \left(\frac{f}{f_{\text{ref}}} \right) \tag{16}$$

- **Raw Chroma Accuracy**: As raw pitch accuracy, except that both the estimated and reference f_0 sequences are mapped onto a single octave. This gives a measure of pitch accuracy which ignores octave errors, a common error made by melody extraction systems:

$$\text{Acc}_{\text{chroma}} = \frac{\sum_\tau v^*_\tau \mathcal{T}\left[\langle \mathcal{M}(f_\tau) - \mathcal{M}(f^*_\tau) \rangle_{12}\right]}{\sum_\tau v^*_\tau} \tag{17}$$

Octave equivalence is achieved by taking the difference between the semitone-scale pitch values modulo 12 (one octave), where

$$\langle a \rangle_{12} = a - 12\lfloor \frac{a}{12} + 0.5 \rfloor. \tag{18}$$

- **Overall Accuracy**: this measure combines the performance of the pitch estimation and voicing detection tasks to give an overall performance score for the system. It is defined as the proportion of all frames correctly estimated by the algorithm, where for non-melody frames this means the algorithm labeled them as non-melody, and for melody frames the algorithm both labeled them as melody frames and provided a correct f_0 estimate for the melody (i.e. within half a semitone of the reference):

$$\text{Acc}_{\text{ov}} = \frac{1}{L} \sum_\tau v^*_\tau \mathcal{T}\left[\mathcal{M}(f_\tau) - \mathcal{M}(f^*_\tau)\right] + \bar{v}^*_\tau \bar{v}_\tau \tag{19}$$

where L is the total number of frames.

The performance of an algorithm on an entire music collection for a given measure is obtained by averaging the per-excerpt scores for that measure over all excerpts in the collection.

2.5.3 Discussion

In the measure definitions provided above, it is assumed that both the estimate and reference sequences are sampled using the same time grid (hop size). In practice, however, this is not always the case, since the time grid of the reference depends on the hop size used by the pitch tracker that produced it, and similarly the time grid of the estimate depends on the specific melody extraction algorithm that produced it. This means that the sequences must be resampled onto a common time-grid prior to the computation of the measures. For the MIREX AME task, any sequence (be it reference or estimate) that is not sampled using a 10 ms hop size is resampled using 0^{th}-order interpolation, i.e. each frequency value in the target sequence is set to its nearest neighbour (in time) from the source sequence. This kind of interpolation can potentially have a detrimental effect on the evaluation, depending on the difference between the source and target time grids. In particular, it can result in artificially low scores for sequences with rapidly changing pitch values, such as opera singing with deep vibrato.

For this reason, the melody evaluator in mir_eval uses 1^{st}-order (linear) interpolation by default in order to map the reference and estimate sequences onto a common time-grid. Assuming the original timestamps of both sequences correspond to the *center* of each analysis frame (as they should), using 1^{st}-order rather than 0^{th}-order interpolation means the results returned by mir_eval are lower-bounded by the MIREX results and are, in our view, more accurate.

2.6 Blind Source Separation

3. COMPARISON TO EXISTING IMPLEMENTATIONS

4. DISCUSSION

5. REFERENCES

[1] Tom Collins. MIREX Task: Discovery of Repeated Themes & Sections, 2013.

[2] Matthew EP Davies, Norberto Degara, and Mark D Plumbley. Evaluation methods for musical audio beat tracking algorithms. Technical Report C4DM-TR-09-06, Centre for Digital Music, Queen Mary University of London, London, England, October 2009.

[3] John E Hopcroft and Richard M Karp. An n^5/2 algorithm for maximum matchings in bipartite graphs. *SIAM Journal on computing*, 2(4):225–231, 1973.

[4] Mark Levy. Improving perceptual tempo estimation with crowd-sourced annotations. In Anssi Klapuri and Colby Leider, editors, *ISMIR*, pages 317–322. University of Miami, 2011.

[5] Mark Levy and Mark Sandler. Structural segmentation of musical audio by constrained clustering. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(2):318–326, 2008.

[6] Hanna M Lukashevich. Towards quantitative measures of evaluating song segmentation. In *ISMIR*, pages 375–380, 2008.

[7] G. E. Poliner, D. P. W. Ellis, A. F. Ehmann, E. Gómez, S. Streich, and B. Ong. Melody transcription from music audio: Approaches and evaluation. *IEEE Trans. on Audio, Speech, and Language Processing*, 15(4):1247–1256, 2007.

[8] J. Salamon, E. Gómez, D. P. W. Ellis, and G. Richard. Melody extraction from polyphonic music signals: Approaches, applications and challenges. *IEEE Signal Processing Magazine*, 31(2):118–134, Mar. 2014.

[9] Douglas Turnbull, Gert RG Lanckriet, Elias Pampalk, and Masataka Goto. A supervised approach for detecting boundaries in music using difference features and boosting. In *ISMIR*, pages 51–54, 2007.