Deliverable D4.10

Evaluation report on the second prototype tool for the automatic semantic description of music samples

Grant agreement nr	688382
Project full title	Audio Commons: An Ecosystem for Creative Reuse of Audio Content
Project acronym	AudioCommons
Project duration	36 Months (February 2016 - January 2019)
Work package	WP4
Due date	31 October 2018 (M30)
Submission date	16 December 2018 (M32)
Report availability	Public (X), Confidential ( )
Deliverable type	Report (X), Demonstrator ( ), Other ( )
Task leader	MTG-UPF
Authors	Frederic Font, Dmitry Bogdanov
Document status	Draft ( ), Final (X)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

Table of contents

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement N° 688382

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

Executive Summary

As part of the Audio Commons Ecosystem, a number of tools are provided for the automatic analysis of audio content without the need for human intervention. These tools are designed for extracting i) musical audio properties for music pieces and music samples, and ii) non-musical audio properties for any kind of sounds. This deliverable addresses the evaluation of the second version of the Audio Commons tool for the automatic analysis of music samples.

For this evaluation we focus on those algorithms included in the annotation tool that provide estimates for specific musical properties. In particular, these algorithms include tempo estimation, key estimation and pitch estimation. In comparison to the previous evaluation of these algorithms already carried out for deliverable D4.4, here we evaluate new versions of each of these algorithms and also include confidence measures. In addition we include the evaluation of the new "single event" algorithm, used to identify audio clips which feature one single sound event (e.g. a single musical note).

We carry out evaluation tasks for each of these properties and compare the algorithm included in the annotation tool with other algorithms described in the literature. To increase the validity of our results, for each of the evaluation tasks we use several datasets which contain content from different sources and different quality levels.

The evaluation results show that the algorithms included in the second version of the Audio Commons annotation tool for music samples tend to perform better than those included in the first version, particularly key estimation and tempo estimation algorithms and when filtering them with confidence measures. However, pitch estimation does not show as much improvement as we would expect after filtering using a confidence measure. During the evaluation we identify a number of aspects for further research that could make both the algorithms and our evaluation better.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

Background

This deliverable is part of the work package WP4 Semantic annotation of musical sound properties. In this work package, we first reviewed the state of the art in automatic extraction of music properties from audio signals (deliverable D4.1), then released and evaluated two tools for the automatic annotation of music samples and music pieces (D4.2, D4.3, D4.4 and D4.5), and then released updated versions of the tool (D4.7 and D4.8). The present deliverable provides an evaluation of the tool described in D4.7. It is recommended to read the present deliverable only after reading deliverable D4.7. By the end of the Audio Commons project, the tool for the automatic annotation of music samples will be updated according to the feedback provided in the present deliverable, yielding deliverable D4.12.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

1 Introduction

1.1 Main objectives and goals

To goal of this deliverable is to assess the effectivity of the automatic annotation algorithms for music samples[1] introduced in the previous deliverable D4.7. To this end, a number of evaluation tasks have been carried out that allow us to estimate the accuracy of the algorithms. The results of this evaluation will inform the next (and last) development phase for the automatic annotation algorithms.

1.2 Methodology

To evaluate the algorithms included in the version of the tool presented in D4.7, we first grouped them into categories (following the grouping proposed in deliverable D4.1Report on the analysis and compilation of state-of-the-art methods for the automatic annotation of music pieces and music), and then ran an evaluation task for the algorithms that generate estimates. The following table shows how the algorithms are grouped, what the basis of their evaluation was and in which section of the present document it can be found. The evaluation tasks we carry out are based on the standards followed on existing sound and music computing literature. At the end of this document, a conclusion section provides a summary of the performance of all of the algorithms.

Name	Description	Group	Evaluation strategy
`ac:duration`	Duration of audio	File metadata	-
`ac:lossless`	Whether audio file is in lossless codec (1 or 0)	File metadata	-
`ac:codec`	Codec used for encoding the audio (e.g. pcm_s16le)	File metadata	-
`ac:filesize`	Size of the file	File metadata	-
`ac:bitrate`	Number of bits per second	File metadata	-
`ac:samplerate`	Number of samples per second	File metadata	-
`ac:channels`	Number of channels	File metadata	-
`ac:audio_md5`	MD5 checksum of raw undecoded audio payload. It can be used as a unique identifier of audio content.	File metadata	-
`ac:tonality`, `ac:tonality_confidence`	Key and scale (e.g. A minor) and confidence of the estimation	Harmony	Accuracy based on ground truth and confidence analysis (Section 2)

[1] In this document this tool is also referred as the “Audio Commons extractor”.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement N° 688382

`ac:loop`	Boolean that indicates whether the audio clip is a loop or not, based on ac:tempo_confidence > 0.95	Rhythm	Same evaluation as for ac:tempo_confidence (Section 2)
`ac:tempo, ac:tempo_confidence`	Tempo in BPM of the audio signal and confidence of the estimation	Rhythm	Accuracy based on ground truth and confidence analysis (Section 3)
`ac:note_name, ac:note_midi, ac:note_frequency, ac_note_confidence`	Played note name (e.g. C4) / Played note midi number (e.g. 60) / Played note frequency (e.g. 440Hz) and confidence of estimations above	Pitch	Accuracy based on ground truth and confidence analysis (Section 4)
`ac:loudness`	Loudness value	Dynamics	-
`ac:dynamic_range`	Dynamic range of audio recording	Dynamics	-
`ac:temporal_centroid`	Temporal centroid	Dynamics	-
`ac:log_attack_time`	Logarithm of the time it takes to reach maximum amplitude of audio signal (good for perceptual attack)	Dynamics	-
`ac:single_evnet`	Whether the audio file contains one single *audio event* or more than one (boolean).		Basic evaluation based on random sampling of datasets and manual assessment (Section 5)
`ac:brightness, ac:hardness, ac:depth, ac:roughness, ac:booming, ac:warmth, ac:sharpness`	Timbral models developed in WP5	Timbre	Evaluated in WP5, D5.7.

1.3 Terminology

AudioCommons: reference to the EC H2020 funded project AudioCommons, with grant agreement nr 688382.

Audio Commons Initiative: reference to the AudioCommons project core ideas beyond the lifetime and specific scope of the funded project. The term "Audio Commons Initiative" is used to imply i) our will to continue supporting the Audio Commons Ecosystem and its ideas after the lifetime of the

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

funded project, and ii) our will to engage new stakeholders which are not officially part of the project consortium.

Audio Commons: generic reference to the Audio Commons core ideas, without distinguishing between the concept of the initiative and the actual funded project.

Audio Commons Ecosystem (ACE): set of interconnected tools, technologies, content, users and other actors involved in publishing and consuming Audio Commons content.

Audio Commons content (ACC): audio content released under Creative Commons licenses and enhanced with meaningful contextual information (e.g., annotations, license information) that enables its publication in the ACE.

Content creator: individual users, industries or other actors that create audio content and publish in the ACE through content providers.

Content provider: services that expose content created by content creators to the ACE.

Content user: individual users, industries or other actors that use the content exposed by content providers and created by content creators in their creative workflows.

Tool developer: individual users, industries or other actors that develop tools for consuming (and also potentially publishing) Audio Commons content.

Embeddable tools: tools for consuming Audio Commons content that can be embedded in existing production workflows of creative industries.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

2 Evaluation of Tempo Estimation

In our evaluation context, the goal of the tempo estimation task is to provide a value in beats per minute (BPM) which matches the periodic rhythmic patterns of a given audio signal. Therefore we might refer to "tempo" or "BPM" interchangeably. Because the tool we are evaluating is aimed at the automatic description of sound samples (as opposed to music pieces), the concept of BPM is mostly relevant in the case of *music loops*. Music loops are typically used in music production environments as building blocks for a music composition. Musicians and producers can combine loops that can be played along together either by selecting loops with the same or compatible musical properties (e.g. tempo, key, genre) or by modifying existing ones. In either case, being able to browse music loops using such properties is crucial for the production process. As was shown in deliverable D2.1: Requirements Report and Use Cases, tempo is one of the most used musical properties for browsing content, therefore good tempo estimation algorithms are crucial for the meaningful annotation of Audio Commons content.

2.1 Datasets

To carry out the tempo estimation evaluation task we use the same datasets that we prepared for the previous evaluation round (deliverable D4.4). This includes three datasets which add up to more than 14k music loops of different production quality, music styles and availability. These are the datasets that we used for the tempo estimation task:

- **Freesound Loops 4000 (FSL4)**: This dataset contains user-contributed loops uploaded to Freesound. It has been built in-house by searching Freesound for sounds with the query terms *loop* and *bpm*, and then automatically parsing the returned sound filenames, tags and textual descriptions to identify tempo annotations made by users. For example, a sound containing the tag *120bpm* is considered to have a ground truth of 120 BPM.

- **Apple Loops (APPL)**: This dataset is composed of the audio loops bundled in Apple's Logic Pro[2] music production software. We parsed the metadata embedded in the audio files using source code available in a public repository[3], and extracted in this way tempo annotations for all the loops. This dataset contains professionally produced loops which are made available in Logic Pro and designed for making music by combining loops from the library. It is the most diverse and complete in terms of music genres, instrumentation, etc.

- **Mixcraft (MIXL)**: This dataset contains all the loops bundled with Acoustica's Mixcraft 7 music production software[4]. Tempo annotations are provided in its loop browser and can be easily exported into a machine-readable format. The dataset aggregates content from different loop sites and is professionally curated by Acoustica.

The following table includes some general statistics about the datasets:

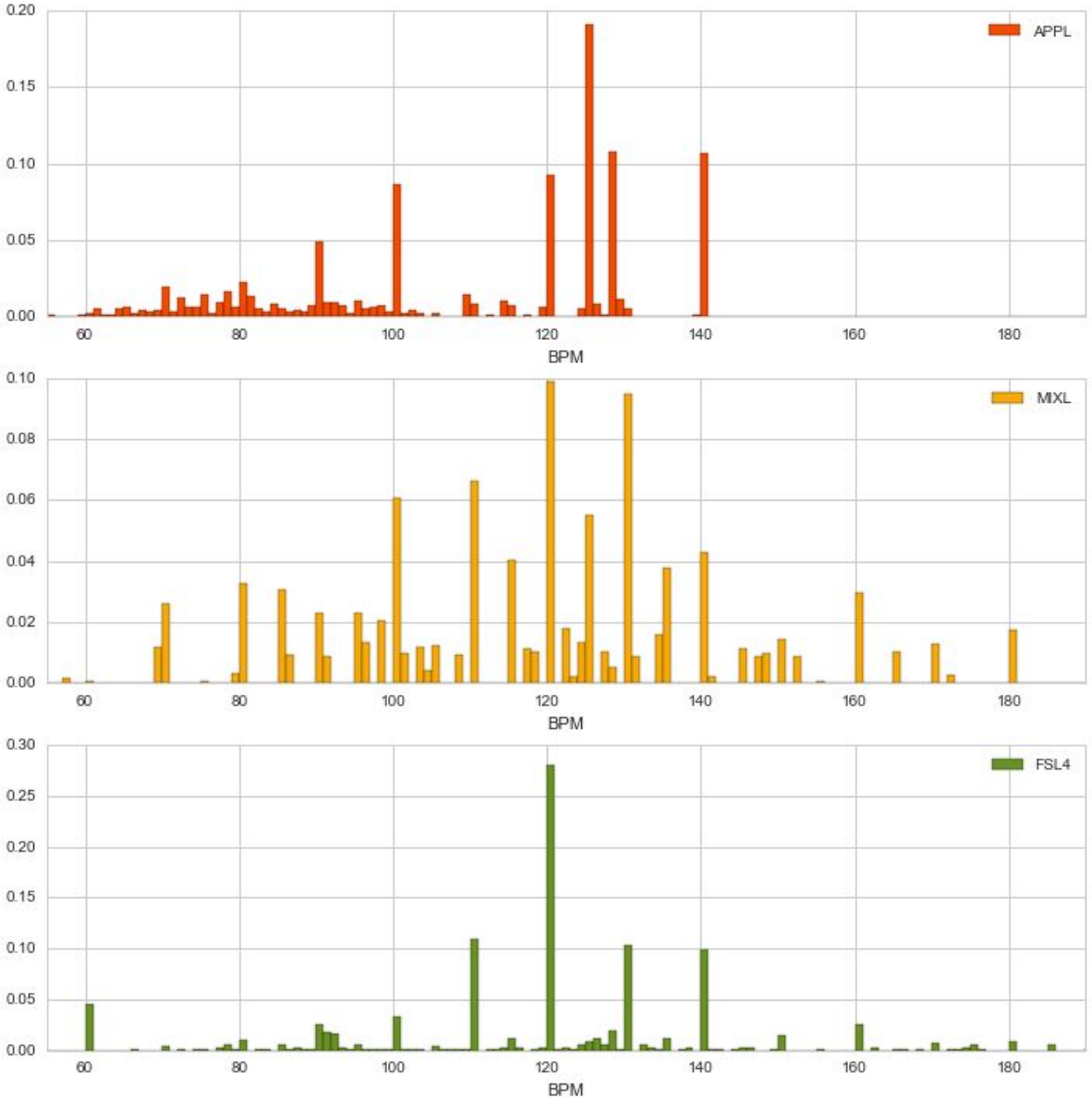
Dataset	N. sounds	Total duration	Mean duration	Duration range
APPL	4611	9h 43m	7.47s	1.32s - 40.05s
MIXL	5451	14h 11m	9.37s	0.32s - 110.77s

[2] http://apple.com/logic-pro
[3] http://github.com/jhorology/apple-loops-meta-reader
[4] http://acoustica.com/mixcraft

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

FSL4	3939	8h 22m	7.63s	0.15s - 30.0s

The plot below shows a histogram of the annotated BPM for each of the datasets. What this histogram tells us is that there are a number of typical BPM values (e.g. 100, 120, 140) which concentrate the majority of sounds in the datasets. Algorithms that are able to correctly estimate BPM in these tempo ranges will perform well in the evaluation stage. Interestingly, the two datasets which come from professional companies (APPL and MIXL) seem to be slightly more evenly distributed than the one coming from Freesound. There are still prominent peaks representing common tempos in those datasets, but sounds are more widely distributed in other tempos as well.



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

2.2 Analysis algorithms

For this second round of evaluation, we included the same algorithms as in the previous evaluation and added the updated tempo estimation algorithm implemented for the second version of the annotation tool:

- **ACExtractorV1**: This is the algorithm included in the first version of the Audio Commons extractor. It uses the tempo estimation method by Degara et. al. [Degara12], which is a probabilistic approach for beat tracking based on inter-onset-interval times and a salience measure for individual beat estimates. This method builds from previous probabilistic beat tracking methods such as Klapuri et. al [Klapuri06]. The final estimated tempo is given based on the mean of estimated beat intervals (see RhythmExtractor2013 algorithm[5]).

- **Percival14**: Percival and Tzanetakis [Percival14] describe a tempo estimation algorithm optimised for low computational complexity that combines several ideas from existing tempo estimation algorithms and simplifies their steps. The algorithm computes an onset strength function based on filtered spectral flux from which tempo lag candidates are estimated using autocorrelation. The most prominent tempo lag is selected and a simple decision tree algorithm is used to chose the octave of the final BPM output. We use a Python implementation of the algorithm provided by the authors in their original paper[6].

- **Böck15**: Böck et. al. [Böck15] propose a novel tempo estimation algorithm based on a recurrent neural network that learn an intermediate beat-level representation of the audio signal which is then feed to a bank of resonating comb filters to estimate the dominant tempo. This algorithm got the highest score in ISMIR 2015 Audio Tempo Estimation task. An implementation by the authors is included in the open-source Madmom audio signal processing library[7].

- **RekBox**: We also include an algorithm from a commercial DJ software, Rekordbox[8]. Details on how the algorithm works are not revealed, but a freely downloadable application is provided that can analyse a music collection and export the results in a machine-readable format.

- **ACExtractorV2**: This is the algorithm included in the second version of the Audio Commons extractor. It is a combination of the method in ACExtractorV1 [Degara12] and a custom re-implementation of the method in [Percival14]. Because in D4.4 we showed that Pervical14 worked the best for music loops, we adopted this algorithm and included a confidence measure for the tempo estimation in the case of audio loops. If the confidence results in a high value, we keep the results of Percival14. However, if the confidence is lower we switch to use the previous method from ACExtractorV1. This is to try to make the algorithm work better for different kinds of music signal. Even though the focus of this section of the deliverable is the analysis of music loops, making the tool a bit more versatile is something that can be seen as a benefit.

[5] http://essentia.upf.edu/documentation/algorithms_reference.html
[6] http://opihi.cs.uvic.ca/tempo
[7] http://github.com/CPJKU/madmom
[8] http://rekordbox.com

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

2.3 Evaluation metrics

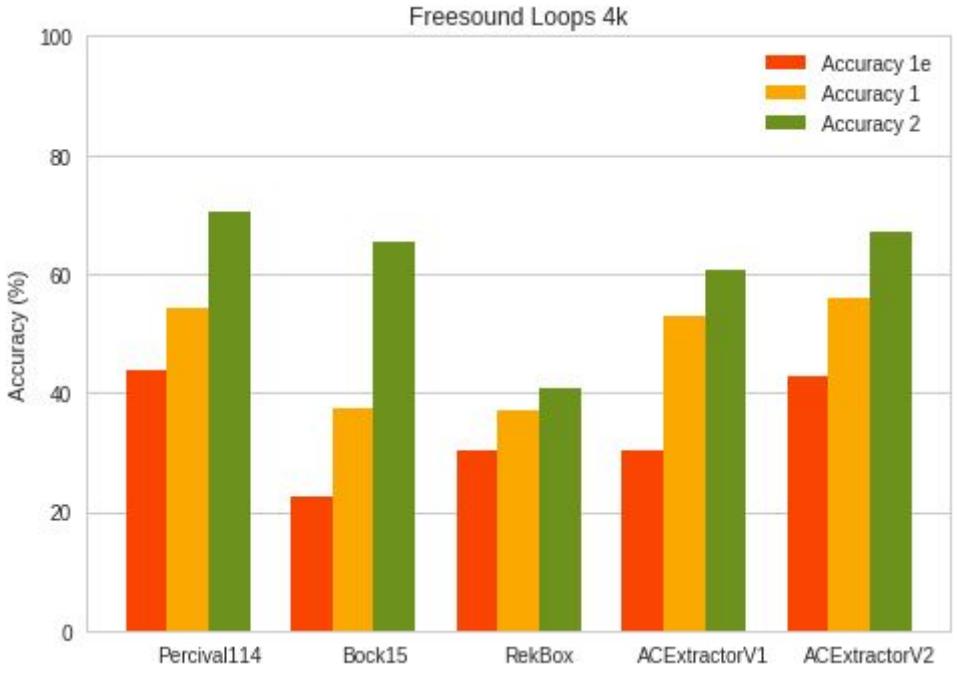
For testing the above algorithms against the three collected datasets we follow standard practice and adopt the methodology described by Gouyon et al. [Gouyon06]. This methodology includes the use of *Accuracy 1* and *Accuracy 2* evaluation metrics. In addition to these metrics, we add an extra measure that we call *Accuracy 1e*. Evaluation metrics are defined as follows:

- **Accuracy1**: percentage of instances whose estimated BPM is within 4% of the annotated ground truth.
- **Accuracy2**: percentage of instances whose estimated BPM is within a 4% of a multiple of ⅓, ½, 1, 2, or 3 times the ground truth BPM.
- **Accuracy1e**: percentage of instances whose estimated BPM is exactly the same as the ground truth after rounding the estimated BPM to the nearest integer.

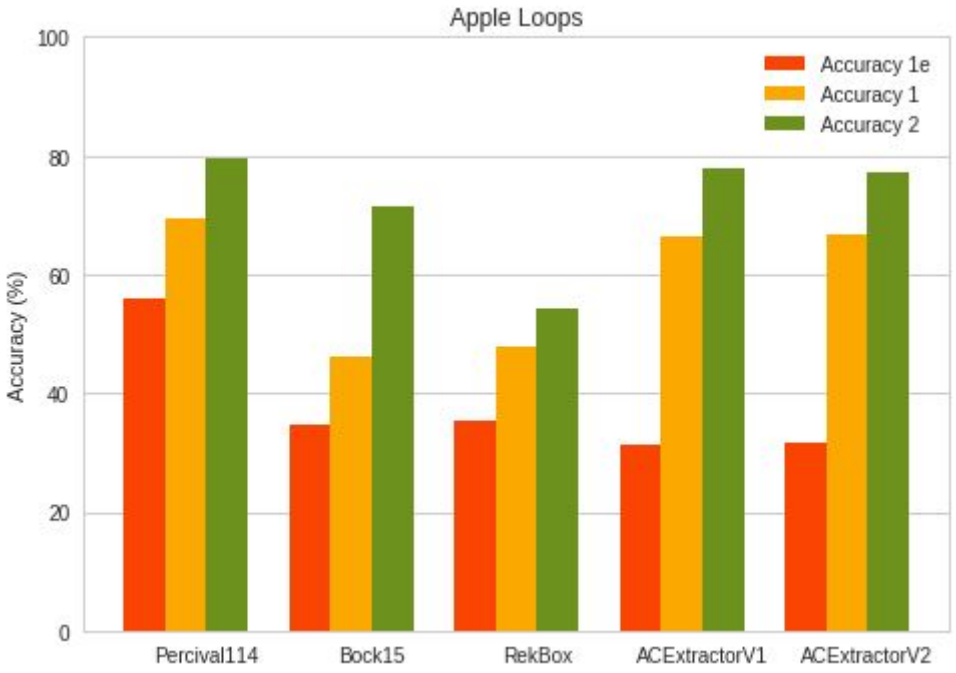
As it can be seen, Accuracy 1e is more strict than Accuracy 1. The reason why we added this extra accuracy measure is that the tool we are evaluating is for the annotation of music samples. Therefore, imagining a music creation context where loops can be queried in a database, it is of special relevance to get returned instances whose BPM exactly matches that specified as target.

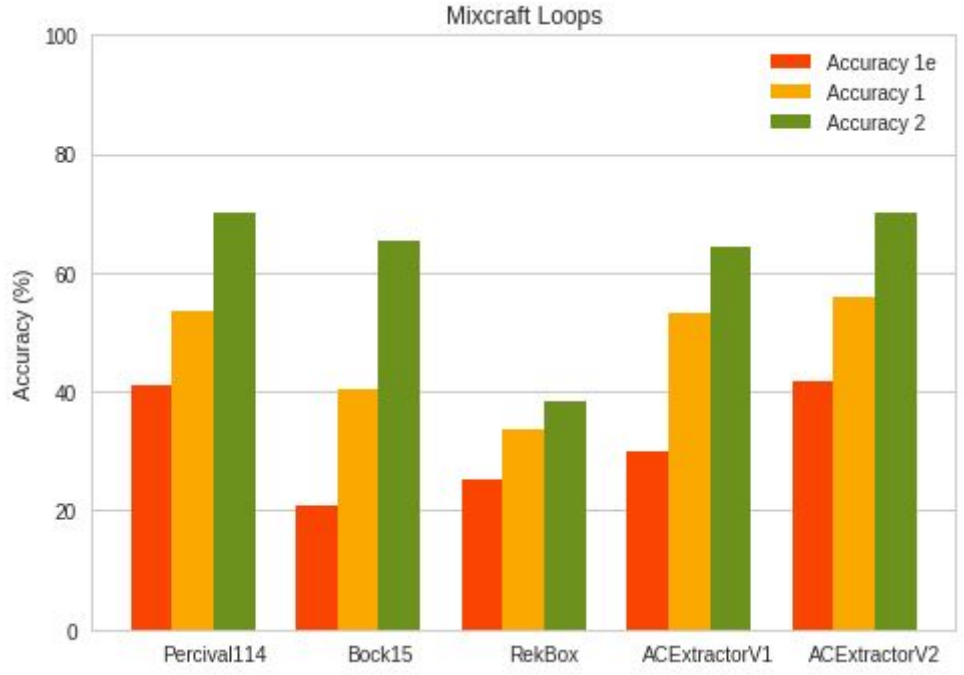
2.4 Results

The following figures show the results of each algorithm in each of the analysed datasets.



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382





The following table shows the average accuracy (in %) when taking into account all datasets at once. Marked in orange are the results of the algorithm included in the Audio Commons tool for the annotation of music samples:

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

Method	Accuracy 1e	Accuracy 1	Accuracy 2	Mean accuracy
Percival14	46.72	59.03	73.29	59.68
ACExtractorV2	38.77	59.57	71.62	56.66
ACExtractorV1	30.46	57.58	67.77	51.94
Böck15	25.93	41.45	67.43	44.94
RekBox	30.01	39.34	44.33	37.89

Overall accuracy results show that either Percival14 or ACExtractor2 always obtain the highest accuracy score for all accuracy measures and all datasets. However, in the case of APPL the accuracies of ACExtractorV2 drop to those of ACExtractorV1. We hypothesize that this is because the step that estimates whether the audio signal is a loop or not does not work well on the APPL dataset (see sections 2.2 and 5) and therefore ACExtractorV2 switches to ACExtractorV1 method. This is what finally makes ACExtractorV2 drop a bit in performance when compared to Percival14.

Considering the data from all datasets at once, mean accuracy values for Percival14 range from 47% (Accuracy 1e) to 73% (Accuracy 2), on average a 3% higher accuracy increase when compared to ACExtractorV2 (the second best-scoring method). With a few exceptions, pairwise accuracy differences between Percival14 and the second best-scored method in all datasets and accuracy measures are statistically significant using McNemar's test and a significance value of $\alpha=0.01$ (i.e., $p \ll 0.01$).

In summary, for this second version of the tool the accuracy of the method included in the AudioCommons extractor has improved significantly. However, for the future version of the analysis tool we should make a decision on whether to keep ACExtractorV2 as it is now or completely switch to Percival14. A complementary evaluation including some non-loop content might be useful and probably favour ACExtractorV2 results (see section 2.2), which would be a reason to stick to ACExtractorV2.

2.5 Analysis of confidence measure

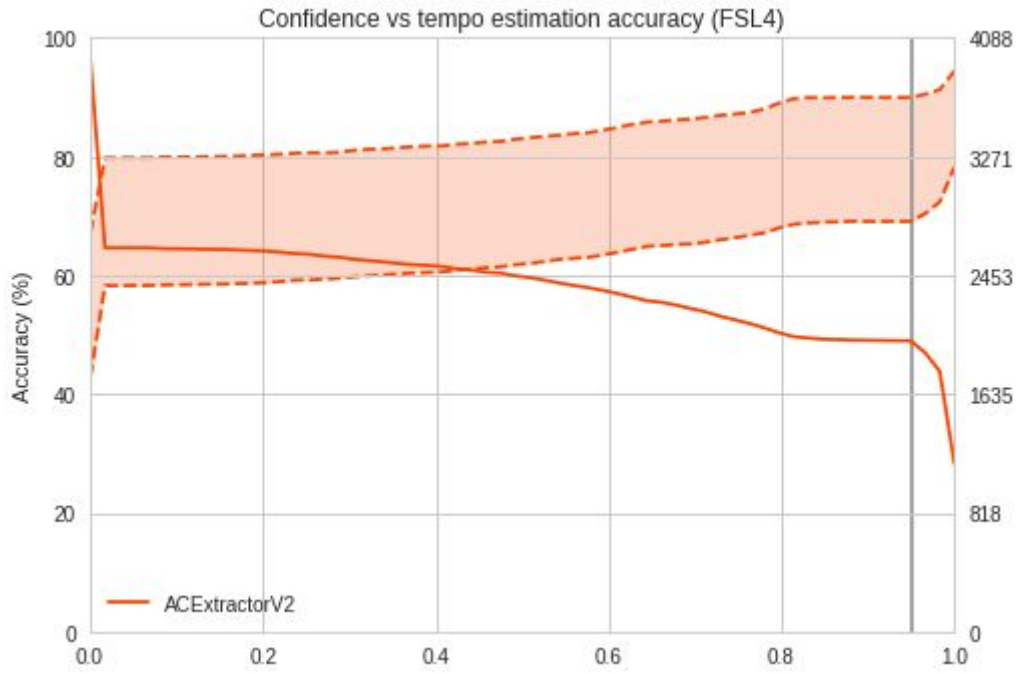
In the second version of the analysis tool a confidence measure for the tempo estimation algorithm was included. The confidence measure is described in [Font16]. It is a simple heuristic that compares the duration of the audio signal with multiples of the beat duration for the estimated tempo. If both durations match well, then it is assumed that the tempo estimation is correct. This heuristic only is applicable to the case of music loops but should return low values for other kinds of signals. This confidence measure works in the range form [0.0 to 1.0].

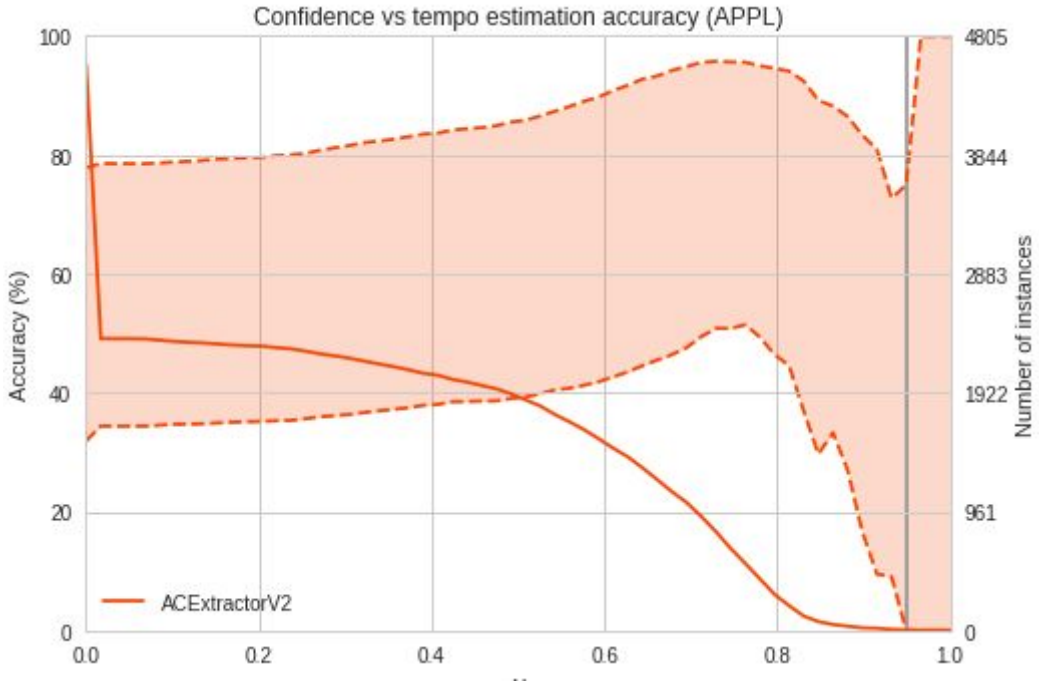
To evaluate the effectivity of the confidence measure we carried out an analysis in which we calculated the accuracy of our estimation method when only including audio clips with the estimated confidence above a threshold. In this way we expect the general accuracy to get higher as minimum confidence gets higher, but we also expect the number of audio clips included in the evaluation (remaining instances in dataset) to get lower as more of them are filtered out. Overall, the idea here is that in a searching interface we can set a minimum estimation confidence to ensure good results and still keep a significant number of results (see deliverable D4.4).

The following figures illustrate the results of our confidence vs tempo estimation analysis. The minimum estimation confidence is indicated in the horizontal axis. The left vertical axis shows the

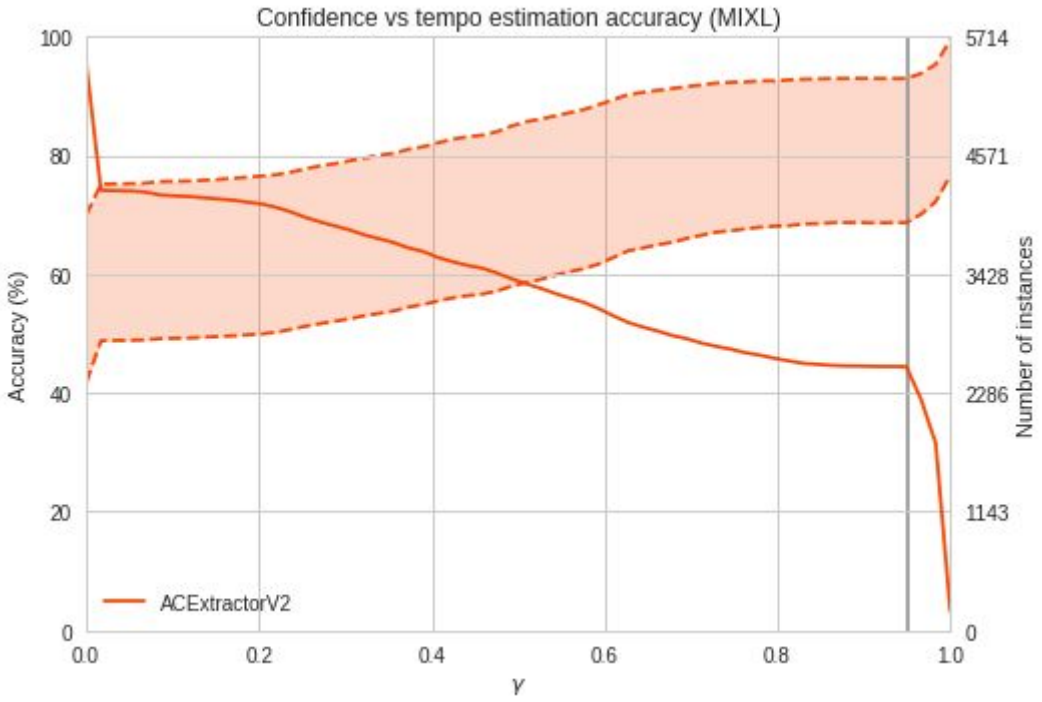
This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

accuracy of the evaluation for a given confidence threshold, the right vertical axis shows the remaining instances in the dataset for a given confidence threshold. Gray vertical line indicates confidence of 95%.





This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382



The table below shows the range of accuracies (from more strict to less strict, A) and the remaining clips in dataset (N) per dataset at the 95% confidence level:

FSL4	**APPL**	**MIXL**
A=69.13 - 89.98%, N=51%	A=0.00 - 75.00%, N=0%	A=68.68 - 92.95%, N=47%

Both in FSL4 and MIXL datasets we can see quite clearly that when confidence gets higher, also accuracies get higher. In both cases for a confidence level of 95% we filter out half of the dataset. In our use case of searching in AudioCommons content this means that we would be able to significantly increase user experience in terms of correctness of results at the expense of returning less results, but we would still return a significant amount.

The case of the APPL dataset behaves a bit differently as accuracy seems to decrease faster with confidence measure and this yields very noisy results for confidence thresholds above 0.75. We hypothesize this is because the method for the confidence estimation does not work very well for audio clips in the APPL dataset. This could be due to a transcoding artifact which adds a bit of silence to the beginning of the clips in APPL, but would need further investigation.

Overall, we see that we with respect to the results shown in 2.4, we can obtain ~25% accuracy increases at the expense of “losing” half of the contents of the dataset.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement N° 688382

3 Evaluation of Key Estimation

According to the aforementioned deliverable D2.1, musical key is another of the most important musical properties that are needed for browsing musical content in a context of music production. Similarly to the case of the evaluation of tempo estimation, here we carry out our evaluation aimed at key estimation for music loops. In our evaluation we assume that the music loops we analyze have a single music key (no significant modulations), and that the key is one of the 24 keys that can be formed with the combination of 12 note roots and two modes (major and minor). This approach is quite common in key estimation evaluation works [Raffel14].

3.1 Datasets

To carry out the tempo estimation evaluation task we use the same datasets that we prepared for the previous evaluation round (deliverable D4.4). In addition, for this deliverable we prepared two new datasets to complement the results (see below). In total we used five datasets which add up to more than 6k music loops/song excerpts of different production quality, music styles and availability. These are the datasets that we used for the key estimation task:

- **Apple Loops (APPL)**: This dataset is composed of the audio loops bundled in Apple's Logic Pro[9] music production software. We parsed the metadata embedded in the audio files using source code available in a public repository[10], and extracted in this way key annotations for all the loops. This dataset contains professionally produced loops which are made available in Logic Pro and designed for making music by combining loops from the library. It is the most diverse and complete in terms of music genres, instrumentation, etc.

- **Mixcraft (MIXL)**: This dataset contains all the loops bundled with Acoustica's Mixcraft 7 music production software[11]. Key annotations are provided in its loop browser and can be easily exported into a machine-readable format. The dataset aggregates content from different loop sites and is professionally curated by Acoustica.

- **GiantSteps Key (GSKY)**: This dataset was compiled in the Giant Steps project [Knees15] and is available through a public Github repository[12]. The dataset contains short fragments of electronic dance music tracks. Music tracks and annotations are extracted from Beatport[13], a digital record store targeted at DJs and focusing on EDM genres. For the audio, only the previews provided by Beatport are used.

- **Apple Loops Reliable (APPL-rel):** This is one of the two new datasets added for this evaluation round. It consists of a small subset of APPL which contains music loops with clear tonality which has been manually verified for us.

- **Mixcraft Reliable (MIXL-rel):** This is one of the two new datasets added for this evaluation round. It consists of a small subset of MIXL which contains music loops with clear tonality which has been manually verified for us.

[9] http://apple.com/logic-pro
[10] http://github.com/jhorology/apple-loops-meta-reader
[11] http://acoustica.com/mixcraft
[12] https://github.com/GiantSteps/giantsteps-key-dataset
[13] http://beatport.com

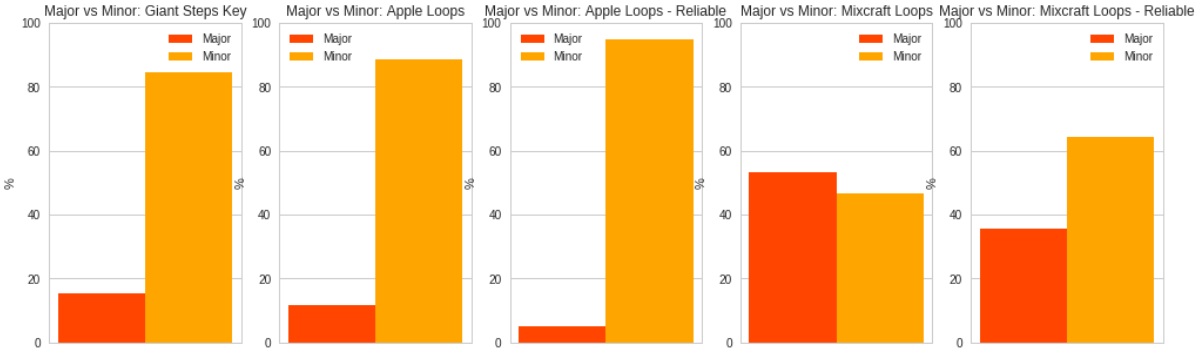
This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

The following table includes some general statistics about the datasets:

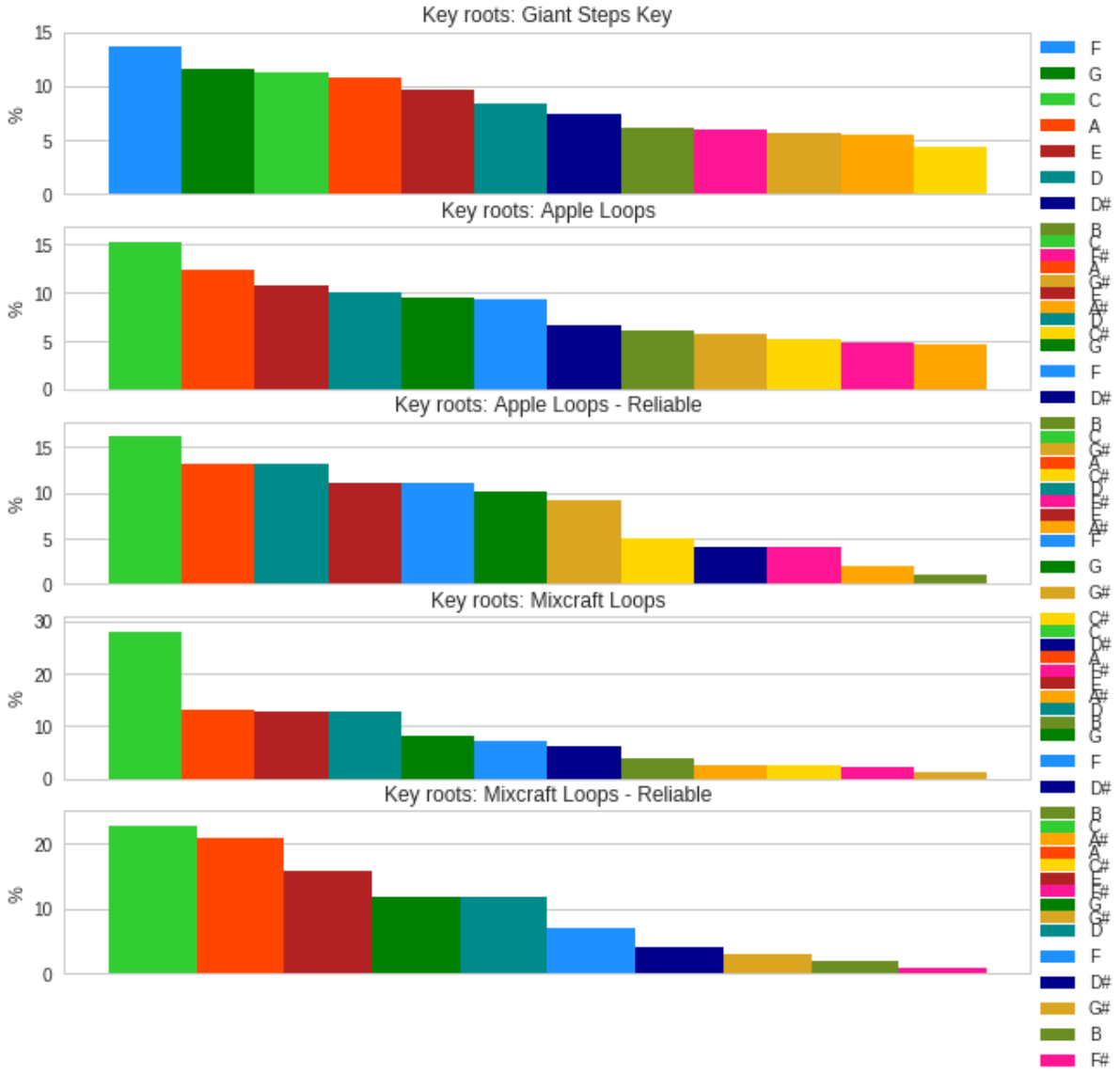
Dataset	N. sounds	Total duration	Mean duration	Duration range
APPL	2743	6h 15m	8.21s	2.07s - 40.05s
APPL-rel	99	0h 13m	8.14s	2.72s - 23.48s
MIXL	2935	9h 37m	11.8s	0.86s - 82.29s
MIXL-rel	101	0h 17m	10.60s	1.81s - 44.65s
GSKY	604	20h 4m	119.61s	44.7s - 120.01s

Note that the sizes of APPL and MIXL are smaller than those for the same datasets in the tempo estimation task. This is because not all of the loops in the datasets contain harmonic instruments that bring a certain key to the loop. The datasets used here are those subsets for which key estimation is provided. Nevertheless, while conducting this evaluation we realized that even though key annotations are provided by APPL and MIXL, these are not always reliable and therefore can bias the estimation results. For this reason we added APPL-rel and MIXL-rel which include only music loops with reliable key annotations and for which we expect the accuracy of key estimation methods to improve significantly.

The plots below show the distribution of major/minor items in the each dataset and the most common roots. Surprisingly enough, both APPL and GSKY have a clear bias for minor tonalities, whereas MIXL is more equally distributed. Same applies for APPL-rel and MIXL-rel. This may be due to the fact that the datasets have different genre coverage (for example, GSKY is focused on electronic dance music, for which minor keys are very common). Regarding key roots, all dataset feature a similar top 5. GSKY features a different ordering in the top 5 roots which can be due to the fact which is the only dataset made out of song excerpts and not loops which were designed for being used as building blocks for other compositions.



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382



3.2 Analysis algorithms

We incorporated the following algorithms in our evaluation of key estimation:

- **ACExtractorV1**: This algorithm included in the first version of the Audio Commons extractor is a standard key estimation algorithm based on HPCP extraction and key profile matching [Gómez06]. Key profiles are extracted from a corpus of European Classical Music [Temperley99]. To provide a single key estimate, HPCP values for each frames are averaged and then compared with the profiles. We use the implementation provided in Essentia, in particular the one included in the FreesoundTonalExtractor[14].

14

https://github.com/MTG/essentia/blob/master/src/essentia/utils/extractor_freesound/FreesoundTonalDescriptors.cpp

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

- **QMULKey**: This algorithm uses the VAMP plugin for Key Detector [Noland07] released by Queen Mary University of London[15]. The Key Detector algorithm analyses a single channel of audio and continuously estimates the key of the music by comparing the degree to which a block-by-block chromagram correlates to the stored key profiles for each major and minor key. The key profiles are drawn from analysis of Book I of the Well Tempered Klavier by J S Bach, recorded at A=440 equal temperament. To provide the overall key we aggregate the result for each frame and take the most repeated (the most common) estimated key.

- **EDMKey#**: This algorithm is developed as an improvement to the existing key estimation method implemented in Essentia (same as in ACExtractorV1) by "introducing a system of multiple profiles and addressing difficult minor tracks as well as possibly amodal ones" [Faraldo17]. The algorithm also performs a HPCP cleaning stage to remove bins with low values which potentially only add noise. The original key profiles used in this algorithm are optimized for electronic dance music (EDM), which also makes it significantly different from the other two methods. In our evaluation we include 3 variations of the EDMKey algorithm which use profiles derived from electronic and popular music (EDMKey1 and EDMKey2) and from Temperley et al. [Temperley99] (EDMKey3).

- **ACExtractorV2**: The updated version of the key estimation algorithm included in the second version of the Audiom Commons extractor consists of a re-implementation of [Faraldo17] for the Essentia analysis library. We expect it to have similar results to EDMKey1 and EDMKey2.

3.3 Evaluation metrics

We evaluate the key estimation task with a number of evaluation metrics which are common in the field. The evaluation metrics are defined as follows:

- **Same**: This metric reports the percentage of key estimations in which both the root and the mode are equal to the annotated ground truth. These are the "exact" matches.

- **Fifth**: This metric reports the percentage of key estimations which have a relation of a perfect fifth with the annotated ground truth.

- **Relative**: This metric reports the percentage of key estimations which have a relation of a relative minor (or relative major) with the annotated ground truth.

- **Parallel**: This metric reports the percentage of key estimations which have a relation of a parallel minor (or parallel major) with the annotated ground truth. That is to say, those in which the root is correctly estimated but the mode is wrong.

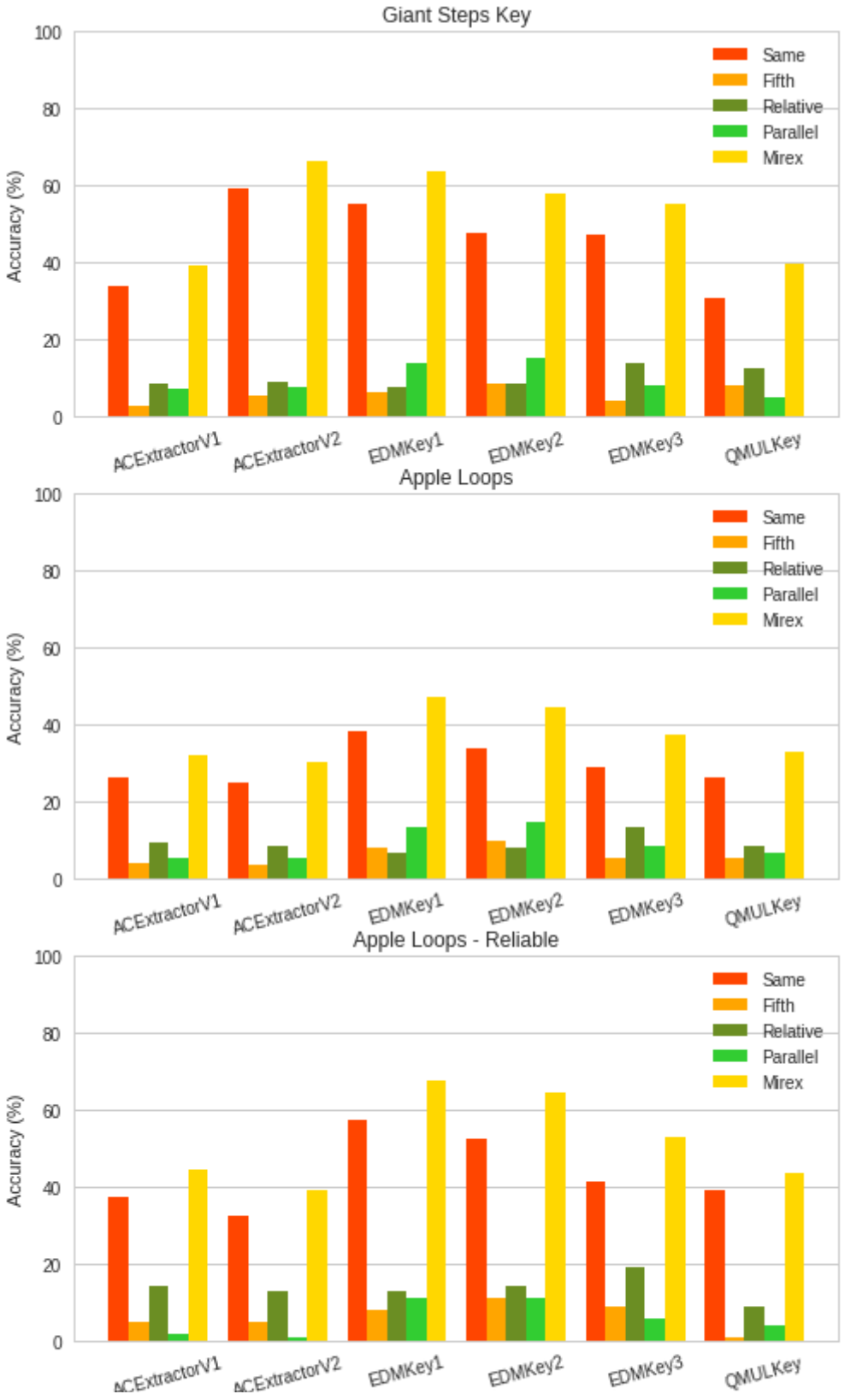
- **Mirex**: This is a combined evaluation metric which computes the ones described above and weights them into a single score. In particular, the "Mirex" score can be computed as *1 * Correct + 0.5 * Fifth + 0.3 * Relative + 0.1 * Parallel*.

3.4 Results

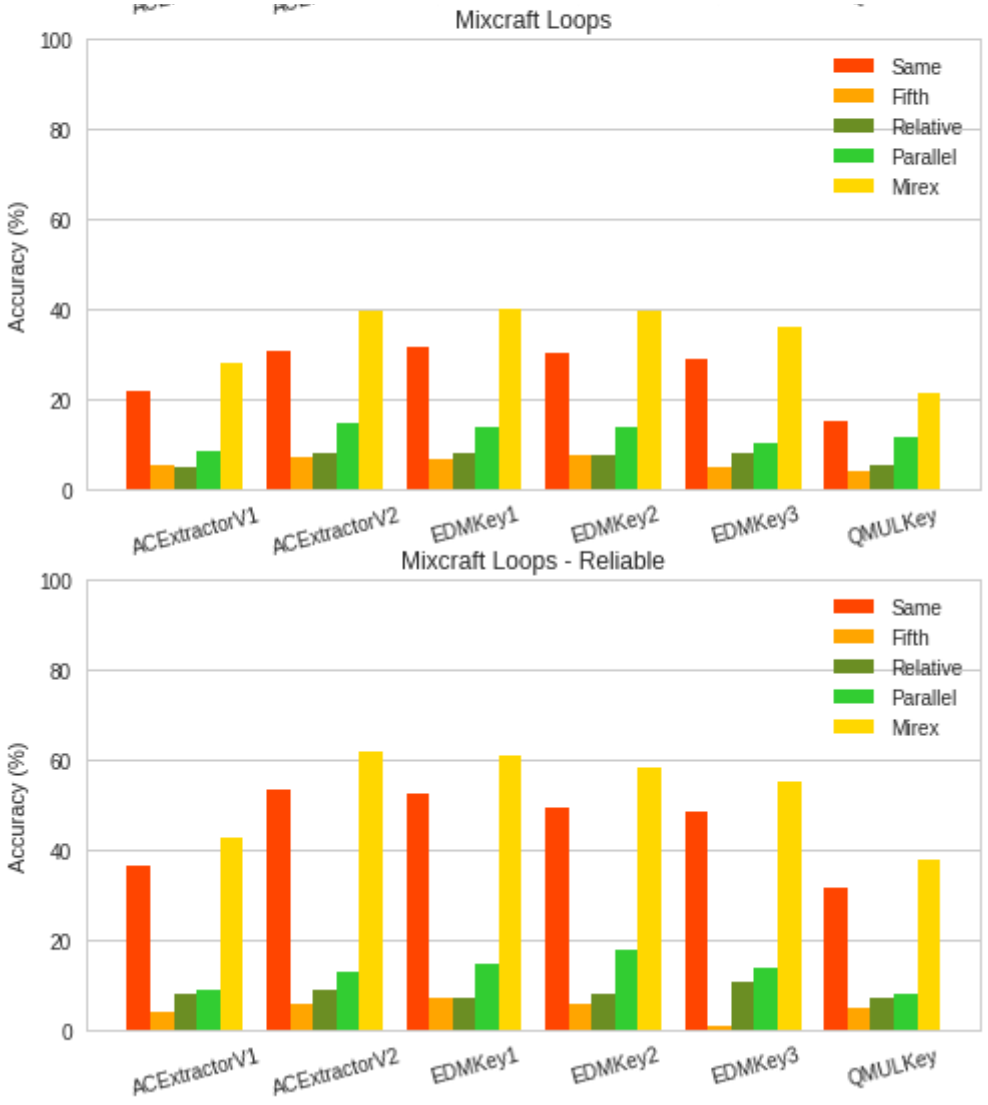
The following figures show the results of each algorithm in each of the analysed datasets.

[15] http://vamp-plugins.org/plugin-doc/qm-vamp-plugins.html#qm-keydetector

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382



The following table shows the average accuracy (in %) when taking into account all datasets at once. Marked in orange are the results of the algorithm included in the second version of the Audio Commons tool for the annotation of music samples:

Method	Same	Fifth	Relative	Parallel	Mirex
EDMKey1	47.13	7.33	8.56	13.41	56.05
EDMKey2	42.88	8.69	9.36	14.58	52.95
ACExtractorV2	40.17	5.42	9.55	8.31	47.41
EDMKey3	39.02	4.97	13.10	9.36	47.31
ACExtractorV1	31.30	4.24	8.97	6.38	37.39
QMULKey	28.67	4.68	8.51	7.08	34.98

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

In general we observe that for all datasets except APPL and APPL-rel, ACExtractorV2 obtains the highest scores with an average increase of ~18% with respect to ACExtractorV1. However, in APPL and APPL-rel ACExtractorV2 performs surprisingly bad, which makes it lower the global accuracy shown in the table above and positions it below EDMKey1 and EDMKey2. This is unexpected as ACExtractorV2 is a re-implementation of EDMKey1 and we expected similar results. This is one aspect that will need to be looked at in more detail.

Another thing to observe is that the newly added *reliable* datasets (APPL-rel and MIXL-rel) feature higher accuracies for all algorithms that their non-reliable counterparts. If we only take into consideration reliable datasets (also including here GSKY, which has all been manually curated), the highest accuracies reached are around 65%.

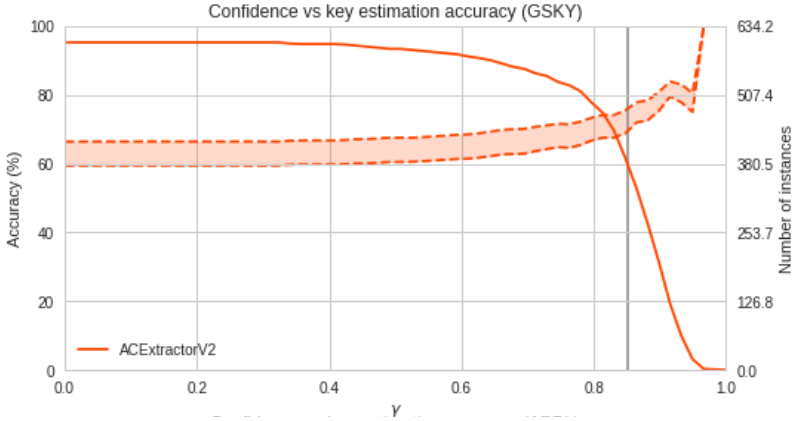
In summary, for this second version of the tool the accuracy of the method included in the AudioCommons extractor has improved significantly. However, there are some problems with the analysis of APPL and APPL-rel dataset which should be further investigated as performances are unexpectedly low and we suspect there might be some non-algorithm related problems (similar to what we suspect is happening with APPL in 2.5 for tempo estimation).

3.5 Analysis of confidence measure

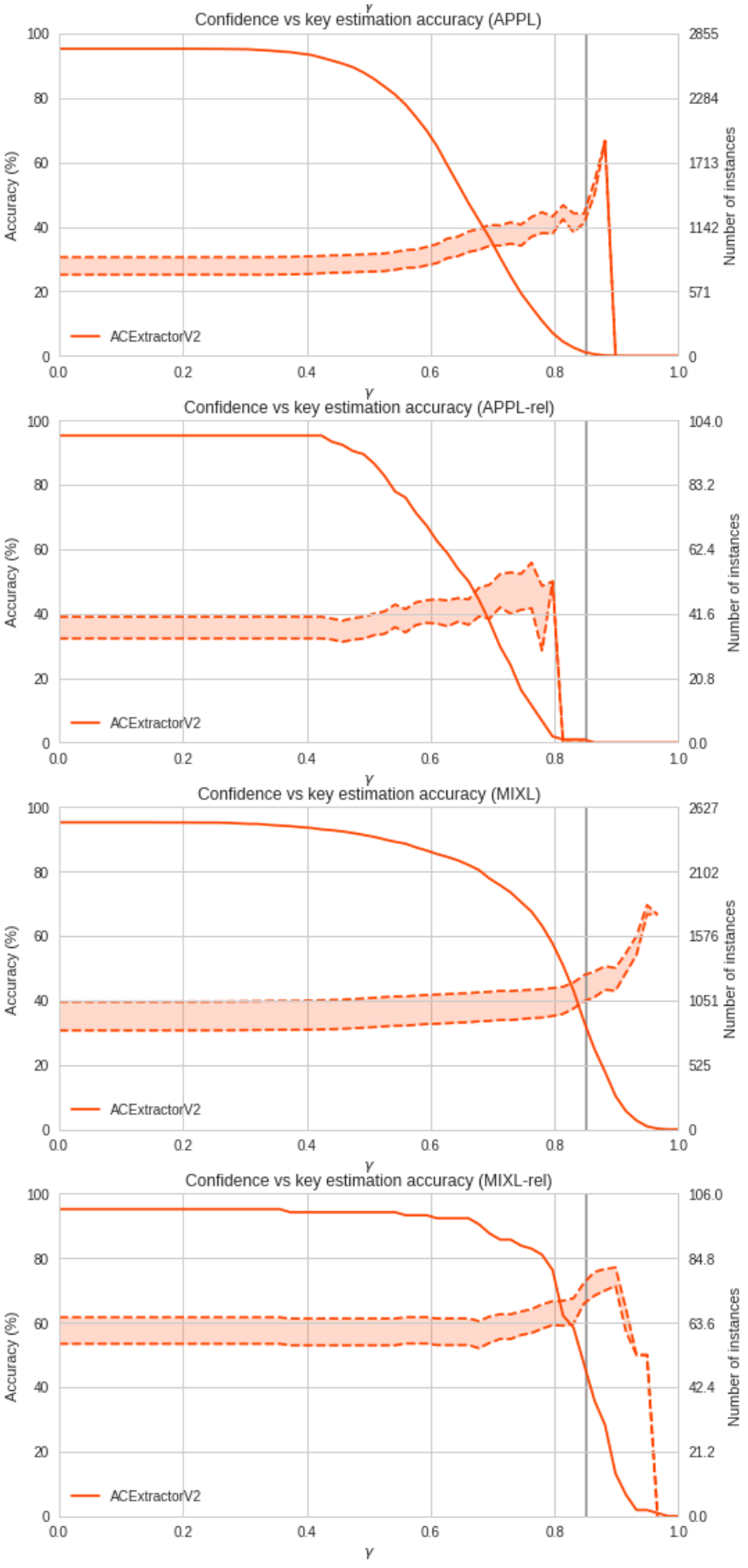
In the second version of the analysis tool a confidence measure for the key estimation algorithm was included. The confidence measure is described in [Faraldo17] and it is based on how well key templates match with the HPCP profiles extracted from the analyzed audio. This confidence measure works in the range form [0.0 to 1.0].

To evaluate the effectivity of the confidence measure we carried out an analysis in which we calculated the accuracy of our estimation method when only including audio clips with the estimated confidence above a threshold. In this way we expect the general accuracy to get higher as minimum confidence gets higher, but we also expect the number of audio clips included in the evaluation (remaining instances in dataset) to get lower as more of them are filtered out. Overall, the idea here is that in a searching interface we can set a minimum estimation confidence to ensure good results and still keep a significant number of results (see deliverable D4.4).

The following figures illustrate the results of our confidence vs key estimation analysis. The minimum estimation confidence is indicated in the horizontal axis. The left vertical axis shows the accuracy of the evaluation for a given confidence threshold, the right vertical axis shows the remaining instances in the dataset for a given confidence threshold. Gray vertical line indicates confidence of 85%.



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

The table below shows the range of accuracies (from more strict to less strict, A) and the remaining clips in dataset (N) per dataset at the 85% confidence level:

GSKY	APPL	APPL-rel	MIXL	MIXL-rel
A=69.27 -75.94% N=64%	A=46.67 - 49.33% N=1%	A=0.00 - 0.00% N=0%	A=39.79 - 47.92% N=34%	A=66.67 - 73.33% N=48%

The first thing that catches our attention is that both APPL and APPL-rel have N~0% when confidence threshold is above 85%. This is similar to what we observed in 2.5 with the analysis of confidence measures for tempo estimation. We suspect there might be some properties of APPL (and APPL-rel) which makes the number of clips remaining in database to go down really quick as confidence threshold increases.

Nevertheless, we can see the tendency of accuracies going up when confidence threshold grows. If we look at the other datasets (and specially the reliable GSKY and MIXL-rel) we observe that we can obtain accuracy increments of ~7% at the expense of "losing" approximately half of the contents of the dataset. This is similar to the results observed for tempo estimation (section 2.5), although the accuracy increases are not as high in this case. One possible way to further improve these results would be to develop more advanced confidence measures that are better able to identify bad estimations, for example by first estimating how predominant is the tonal content of an audio clip.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

4 Evaluation of Pitch Estimation

In the previous evaluation tasks we focused on musical properties that make sense for music fragments long enough to introduce a notion of tempo and tonality. In this case we focus the evaluation in another relevant musical property for shorter music sounds. In particular we look into pitch estimation for recordings of single notes from pitched musical instruments. This is important as proper pitch annotation allows the reuse of these sounds in, for example, a music sampler. In our context of content reuse within a creative musical context, more than the pitch we are interested in estimating the musical note which is played in a single note recording. For this reason, we evaluate algorithms which are capable of outputting a MIDI note number (and a note name) along with the non-quantized pitch value (see below).

4.1 Datasets

To carry out the pitch estimation evaluation task, we use the five datasets that were already introduced in the previous evaluation (deliverable D4.4), and add a new extra dataset. This adds up to more than 30k instrument notes from different instruments and recording conditions. These are the datasets that we used for the pitch estimation task:

- **Freesound, Carlos Vaquero dataset (CVAQ)**: This dataset contains single-note recordings from different western music instruments containing, among others, acoustic guitar, recorder, bassoon and several bowed instruments (played plucked and bowed with different techniques). This dataset was created in the Music Technology Group of Universitat Pompeu Fabra. All these sounds are hosted in Freesound and released under Creative Commons Licenses.

- **Freesound, Good-Sounds dataset (GSND)**: Similarly to CVAQ, this dataset contains single-note recordings from different western music instruments but only played with their most common playing technique. This dataset was created in the Music Technology Group of Universitat Pompeu Fabra. Again, all these sounds are hosted in Freesound and released under Creative Commons Licenses.

- **University of IOWA (IOWA)**: This is a classic dataset for pitch estimation which includes single notes from piano, cello, trumpet, marimba and xylophone, and use different playing techniques. This dataset has been developed at the Electronic Music Studios of University of Iowa[16].

- **Philharmonia Orchestra Sound Samples (PHIL)**: The Philharmonia Orchestra Sound Samples dataset is composed of single-note recordings of a wide range of orchestral instrument which were recorded and made available under a Creative Commons license by the London Philharmonia Orchestra[17]. For this particular dataset, we discarded recordings belonging to percussion instruments which are typically non-pitched.

- **NSynth test dataset (NSYT)**: NSynth dataset is a large-scale dataset of musical notes released by Google as part of the Magenta project [Engel17]. The dataset is released under a Creative Commons license and can be freely downloaded online[18]. The full version of NSynth contains more than 300k pitched musical notes generated using instrument plugins from commercial sample libraries. NSYT only includes the subset of the dataset labeled as “test”..

[16] http://theremin.music.uiowa.edu/mis.html
[17] http://www.philharmonia.co.uk/explore/sound_samples
[18] https://magenta.tensorflow.org/datasets/

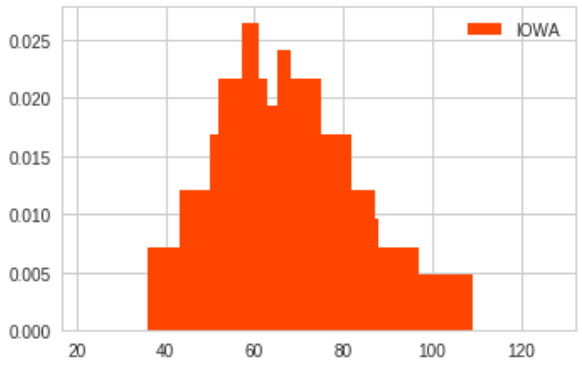
This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement N° 688382

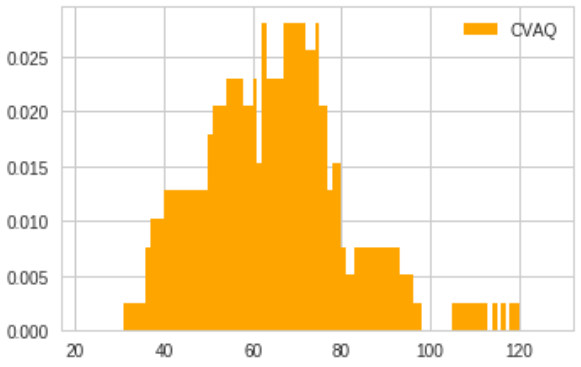
- **NSynth test dataset (NSYV)**: NSynth dataset is a large-scale dataset of musical notes released by Google as part of the Magenta project [Engel17]. The dataset is released under a Creative Commons license and can be freely downloaded online[19]. The full version of NSynth contains more than 300k pitched musical notes generated using instrument plugins from commercial sample libraries. NSYT only includes the subset of the dataset labeled as "validation". This is a new dataset included for this deliverable.

The following table includes some general statistics about the datasets:

Dataset	N. sounds	Total duration	Mean duration	Duration range
IOWA	415	0h 18m	2.68s	0.14s - 7.73s
CVAQ	391	0h 23m	3.62s	1.01s - 8.28s
PHIL	5462	2h 26m	1.61s	0.08s - 45.61s
NSYT	4096	4h 33m	4.0s	4.00s - 4.00s
NSYV	12678	14h 5m	4.0s	4.00s - 4.00s
GSND	8141	14h 24m	6.45s	0.48s - 33.65s

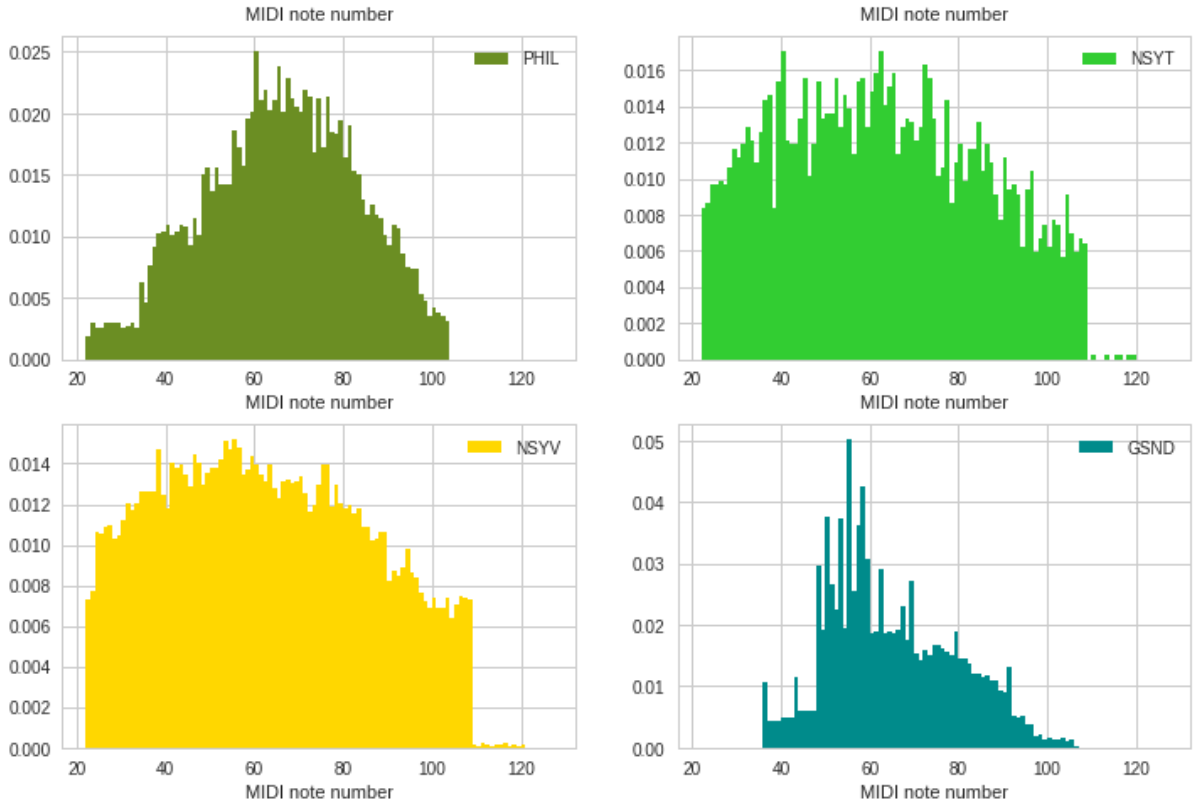
The plots below show a histogram of the MIDI note numbers featured in each of the datasets. We can see how NSYT and NSYV show the most uniform distribution. This is due to the fact that sounds in NSYT/NSYV were generated through an automated process that played all possible notes of every instrument using software plugins and recorded the output. In this way it was possible to easily generate notes for the entire spectrum. The other datasets (which contain similar sets of western classical music instruments) feature a similar histogram. This suggests that NSYT/NSYV dataset might include notes from instruments played in ranges which would not be played by such instruments very often.





[19] https://magenta.tensorflow.org/datasets/

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382



4.2 Analysis algorithms

We incorporated the following algorithms in our evaluation of pitch estimation:

- **ESSPYin**: This is a classic implementation of the well-known YIN pitch estimation algorithm [Cheveigné02]. For this algorithm we use the implementation provided in Essentia[20]. Pitch YIN is based on autocorrelation of the signal and further peak detection to estimate the fundamental frequency. This algorithm outputs an estimated pitch for each frame of the audio. The final selected pitch is taken as the median of all these pitches. For our analysis, the output provided by the algorithm is translated into the (quantised) MIDI note number using the convention that A4 = MIDI 60 = 440hz.

- **QMULPYin:** This is an implementation of a variant of the YIN algorithm, probabilistic YIN [Mauch14], provided as a VAMP plugin from Queen Mary University of London. This algorithm outputs an estimated pitch for each frame of the audio. The final selected pitch is taken as the median of all these pitches. Again, the output provided by the algorithm is translated into the (quantised) MIDI note number using the convention that A4 = MIDI 60 = 440hz.

- **ACExtractorV1**: This is the algorithm that's included in the first version of the Audio Commons extractor. The Audio Commons extractor uses the algorithm pitch YIN FFT for pitch estimation, which is an optimisation proposed by Brossier [Brossier06] for reduced calculation time of the original YIN. This algorithm outputs an estimated pitch for each frame of the audio. Again, we use the implementation provided in Essentia[21], and the median value

[20] http://essentia.upf.edu/documentation/reference/std_PitchYin.html
[21] http://essentia.upf.edu/documentation/reference/std_PitchYinFFT.html

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

of the output provided by the algorithm for each frame is translated into the (quantised) MIDI note number using the convention that A4 = MIDI 60 = 440hz.

- **ACExtractorV2**: This is the algorithm that's included in the second version of the Audio Commons extractor. For pitch estimation, we are using the same algorithm as in the first version of the extractor as the performance was already very similar to that of state of the art algorithms. Therefore, ACExtractorV2 is the same algorithm as ACExtractorV1 and should report same results.

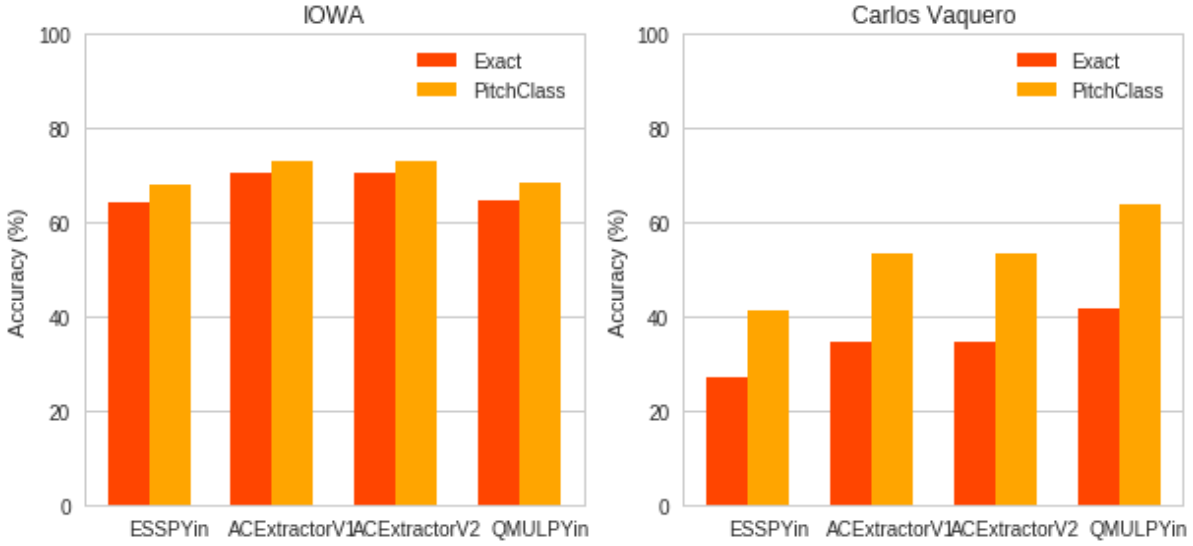
4.3 Evaluation metrics

To evaluate the effectiveness of the pitch estimation methods we use the following two accuracy metrics:

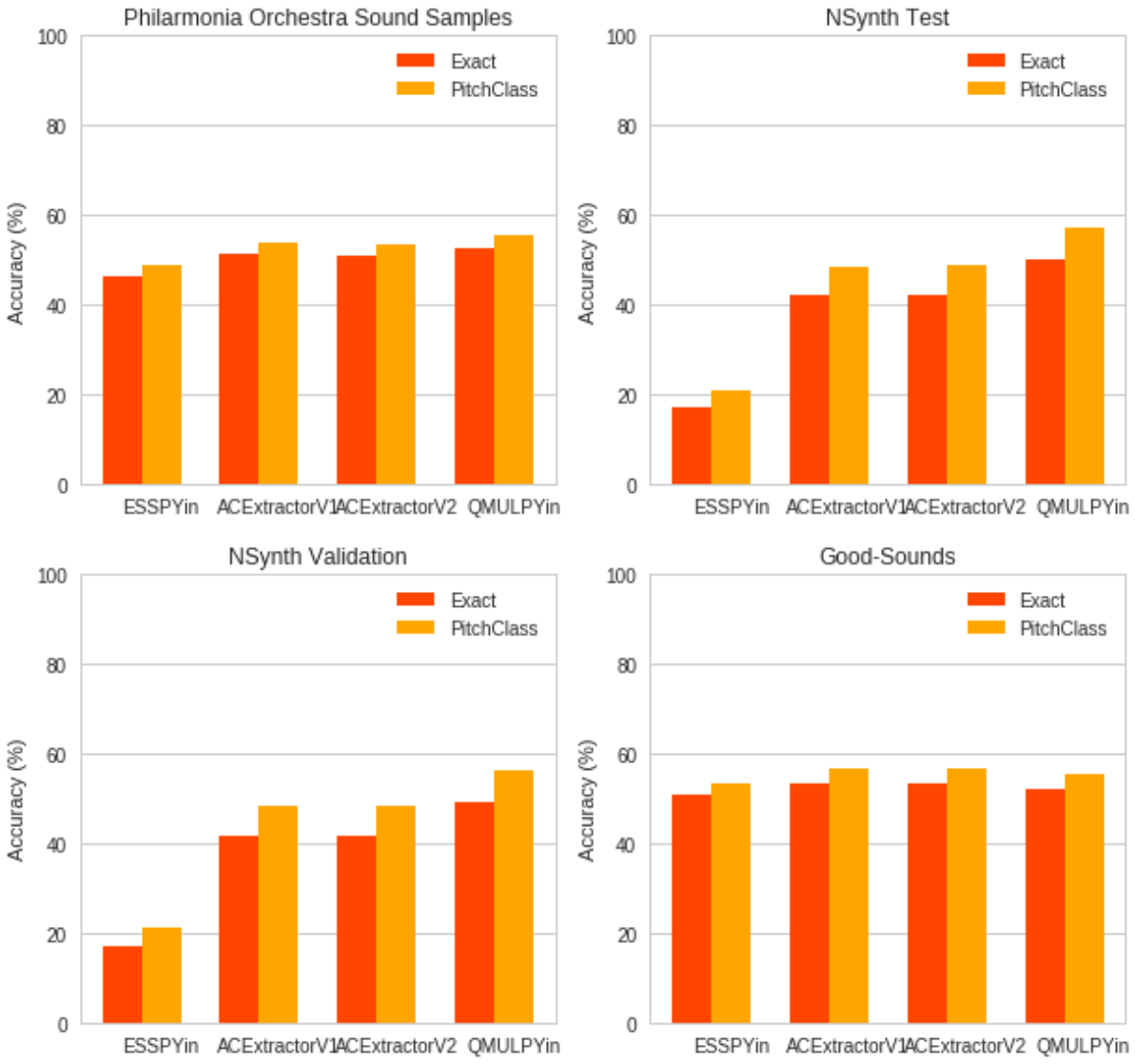
- **Exact**: This metric only considers an estimation to be correct when it's the same as the annotation in the ground truth. As mentioned above, our analysis algorithms output integer MIDI notes. These are the values that we use for comparison, therefore there is no tolerance threshold. We assume all algorithm outputs and the dataset ground truth annotation use the convention that A4 = MIDI 60 = 440hz.
- **PitchClass**: This metric differs from the "Exact" one in that it also considers as correct those estimations which feature the same pitch class as the ground truth. This means that octave errors are not considered wrong in this metric.

4.4 Results

The following figures show the results of each algorithm in each of the analysed datasets.



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382



The following table shows the average accuracy (in %) when taking into account all datasets at once. Marked in orange are the results of the algorithm included in the Audio Commons tool for the annotation of music samples:

Method	Exact	PitchClass	Mean
QMULPYin	51.72	59.50	55.61
ACExtractorV2	48.82	55.59	52.20
ACExtractorV1	48.85	55.56	52.20
ESSPYin	37.10	42.29	39.69

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

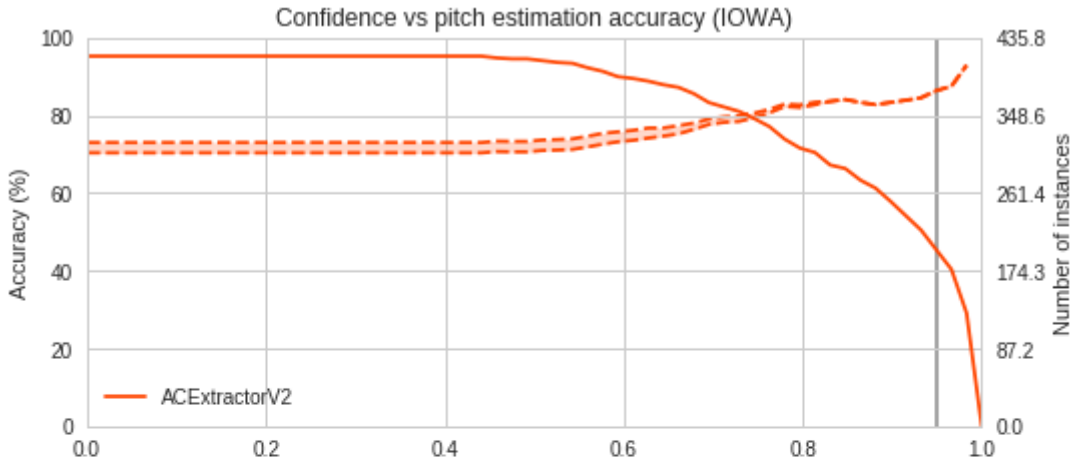
It can be observed that, as expected, ACExtractorV2 reports the same results as ACExtractorV1 and that these are similar to the best-scoring method, QMULPYin. Nevertheless, it is quite unexpected to observe such big differences among the results obtained using different datasets, specially in NSYT and NSYV in which results ACExtractorV1/V2 are lower. Still, the overall accuracies are rather low and it should be possible to improve them by inspecting typical errors and understanding what is it making the algorithms fail.

4.5 Analysis of confidence measure

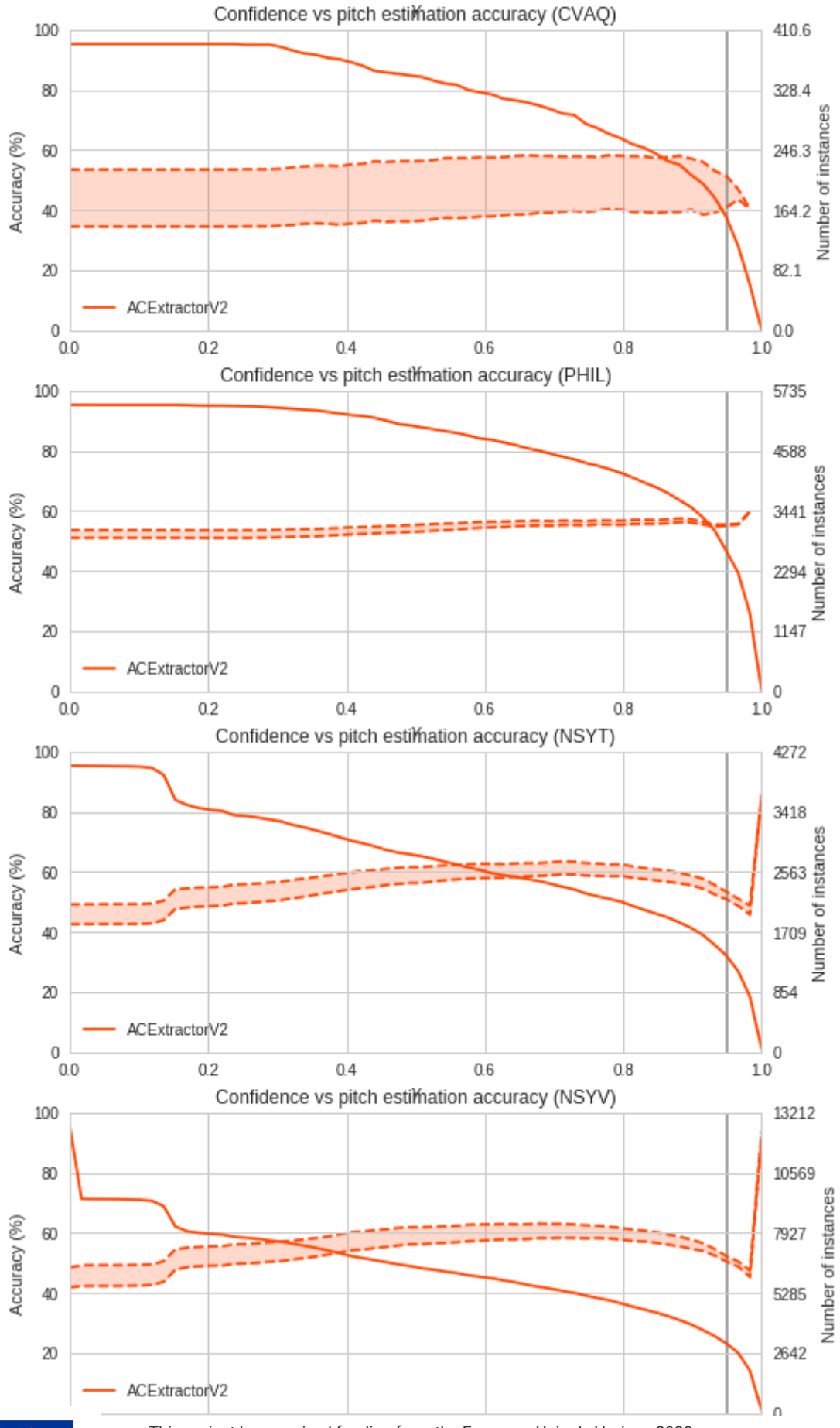
In this second version of the analysis we introduce a confidence measure for pitch estimation as described in [Brossier06] and implemented in the Essentia audio analysis library. This confidence measure works in the range form [0.0 to 1.0].

To evaluate the effectivity of the confidence measure we carried out an analysis in which we calculated the accuracy of our estimation method when only including audio clips with the estimated confidence above a threshold. In this way we expect the general accuracy to get higher as minimum confidence gets higher, but we also expect the number of audio clips included in the evaluation (remaining instances in dataset) to get lower as more of them are filtered out. Overall, the idea here is that in a searching interface we can set a minimum estimation confidence to ensure good results and still keep a significant number of results (see deliverable D4.4).

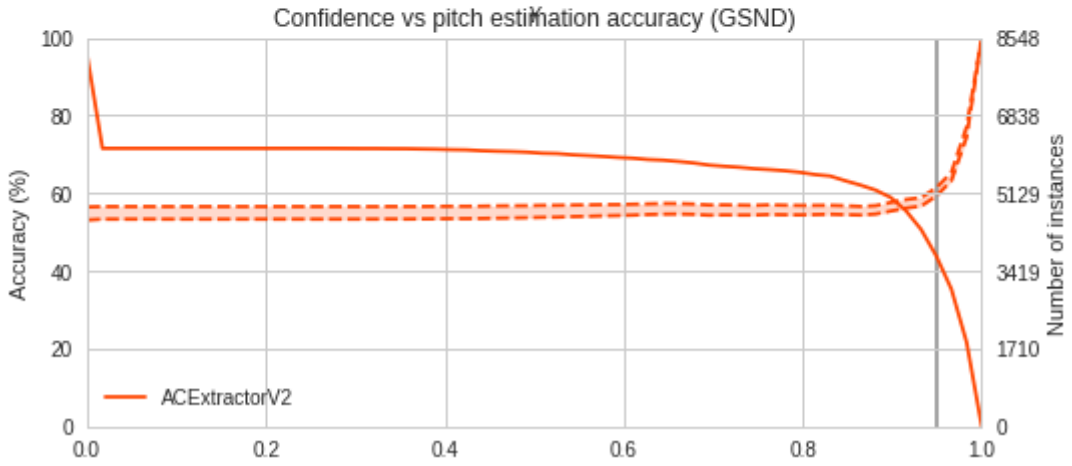
The following figures illustrate the results of our confidence vs pitch estimation analysis. The minimum estimation confidence is indicated in the horizontal axis. The left vertical axis shows the accuracy of the evaluation for a given confidence threshold, the right vertical axis shows the remaining instances in the dataset for a given confidence threshold. Gray vertical line indicates confidence of 95%. Dotted lines show accuracy for different evaluation measures from less strict (top) to more strict (bottom).



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382



The table below shows the range of accuracies (from more strict to less strict, A) and the remaining clips in dataset (N) per dataset at the 95% confidence level:

IOWA	CVAQ	PHIL	NSYT	NSYV	GSND
A=86.36 - 86.36% N=48%	A=40.52 - 50.98% N=39%	A=55.03 - 55.44% N=49%	A=50.66 - 53.16% N=33%	A=50.26 - 52.04% N=24%	A=59.52 - 61.59% N=46%

What we observe in this case is that only IOWA and, to a less extent, GSND datasets behave in the expected way and and show significant average accuracy increases (10% and 5% respectively) when we raise the confidence threshold (again, at the expense of "losing" approximately half of the dataset). For the other datasets we observe unexpected behaviours in which there seem to be some accuracy improvements together with confidence threshold but after a certain threshold accuracy goes down again. We currently don't have an explanation for that behaviour but it should be further examined.

Overall, the confidence measure for pitch estimation does not work as expected in most of the cases and the accuracy curves observed in most of the datasets suggest that the confidence measure we used is not a good proxy for the reliability of the pitch estimation. This is one aspect that should be further investigated in future iterations of the tool.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

5 Evaluation of "single event" descriptor

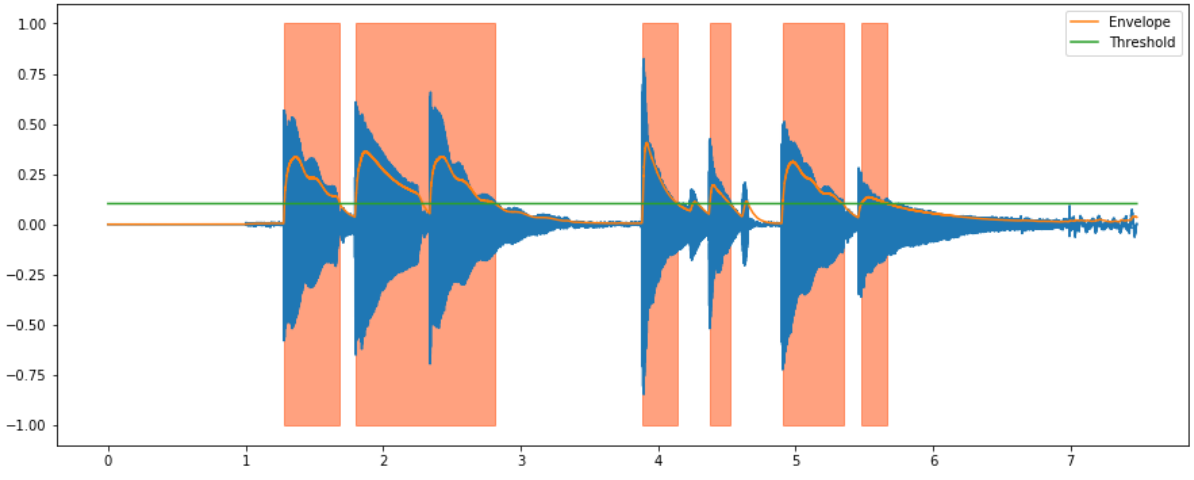
While developing ACExtractorV2 we included a new descriptor called "single event" that is able to separate between audio files which consists of a single sound event (i.e. a single energy burst) from other kinds of more complex sound events. This is useful as a sort of confidence measure to detect for which audio files the descriptors which are better suited to single events will potentially be more meaningful. This includes timbral models developed in WP5 and other descriptors present in ACExtractorV2 such as pitch estimation and log attack time. In this section we outline the new descriptor we developed and evaluated with a small manually crafted dataset.

5.1 Single event detector

The single event descriptor algorithm has the following steps:

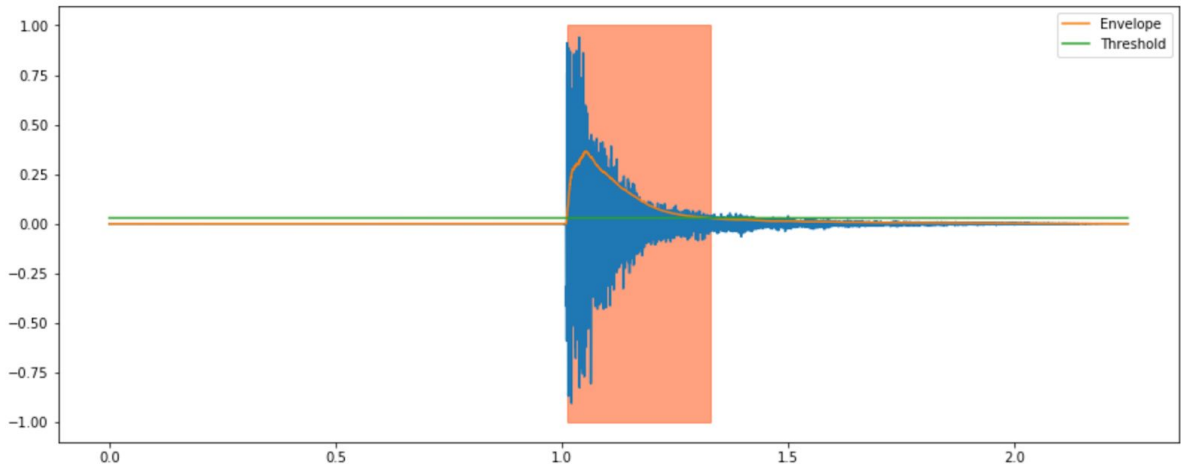
1. Compute energy envelope of the signal.
2. Set a "silence threshold" as a multiple of the average energy from the envelope.
3. Identify potential event regions. Regions are defined as signal parts between the envelope going above the silence threshold and then going below.
4. Join together contiguous regions which are very close in time.
5. Discard regions whose energy is below an energy region threshold.
6. If there is only one region, the sound is considered a single event.

The figure below exemplifies the different steps of the algorithm. We can observe regions marked in orange in the sections where energy envelope goes above the silence threshold. The regions corresponding to the 2nd and 3rd visible signal bulbs in the sound (in seconds 2 to 3 approximately) have been collapsed into a single region as the second bulb starts very shortly after the first. Also, small energy bulbs happening at seconds 4.2 and 4.6 (approximately) are not considered valid regions as their energy is too low. In this example, our algorithm would consider this sound not to be single event as multiple regions are detected.



In this second example the sound contains a single event which is properly detected by the algorithm and labelled as such.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382



Note that this algorithm is only based in signal energy and does not take into account changes in frequency which might correspond to different events in a single recording. For example, the recording of a violin playing a musical scale in *staccato* will result in each note being identified as a different event, while the same scale being played in *legato* (i.e. there is no silence between notes) might result in the whole scale being detected as a single event. To account for cases like this, future versions of this algorithm can include other signals besides the energy envelope in which to detect candidate regions. The specific values used for the thresholds and implementation of the algorithm can be found on the source code repository[22].

5.2 Evaluation

We conducted a simple evaluation task in which we manually labelled 224 examples of Freesound sounds as either being *single event* or *multiple event*.The 224 examples were randomly selected. The resulting dataset contains 149 examples of multiple event sounds and 75 examples of single event sounds. Considering this dataset we run our single event detection algorithm and obtain a classification accuracy of 65% which can be decomposed as in the following table:

True positives (single event sounds classified as single event)	25%
True negatives (multiple event sounds classified as multiple event)	40%
False positives (multiple event sounds classified as single event)	27%
False negatives (single event sounds classified as multiple event)	8%
Global accuracy True positives + True negatives	65%

22 https://github.com/AudioCommons/ac-audio-extractor/blob/aabdf0d2a3242303ec461f986b856895ee54db55/analyze.py#L51-L135

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

Interestingly enough most of the errors come from the “false positives” category, meaning that sounds with multiple events are classified as being single event. For future improvements in the algorithm we should further investigate why we obtain such high rates of false positives, but we suspect that our parameters for collapsing contiguous regions might need to be more strict.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement N° 688382

6 Conclusion

In this deliverable we have described the evaluation tasks that we have carried out for the most important estimation algorithms included in the second version of the Audio Commons tool for the automatic annotation of music samples. The following table summarises the accuracies obtained for each of the tasks:

Task	**Accuracy (for the AC tool v1)**	**Accuracy (for the AC tool v2)**	**Accuracy (for the AC tool v2 + confidence measures)**
Tempo estimation	30% - 67%	39% - 72%	69% - 93%
Key estimation	23% - 28%	40% - 65%	40% - 75%
Pitch estimation	50% - 54%	49% - 56%	50% - 86%
Single event	-	65%	-

As it can be observed, the algorithms included in the second version of the Audio Commons annotation tool for music samples tend to perform better than those included in the first version, particularly key estimation and tempo estimation algorithms and when filtering them with confidence measures. However, pitch estimation does not show as much improvement as we would expect after filtering using a confidence measure. During the evaluation we identified a number of aspects for further research that could make both the algorithms and our evaluation better. For example, we observed that APPL dataset features some potential audio issues and mislabelled key annotation (this also applies to MIXL). Nevertheless comparison with state of the art algorithms reveals that AC descriptors feature similar performance. An interesting future evaluation step would be the evaluation with real users to assess whether the accuracies obtained after filtering using confidence measures are enough for a good user experience when searching and browsing Audio Commons content.

Besides the improvements in the tempo, key and pitch estimation algorithms we also included a “single event” descriptor that can tell apart audio files containing one single sound event. This can be useful to determine when the application of some of the algorithms included in the AC extractor is more relevant. For example, pitch estimation is not relevant on a music loop and tempo estimation is not relevant for a single note. This descriptor can be used in combination with the confidence measures to further filter non-relevant content for a given descriptor.

Overall we conclude that this second version of the AC audio analysis tool includes several improvements with respect to the previous version. It is worth mentioning here that we already used this version of the tool to analyze most of the contents of Freesound and made the results available through the Freesound API and AudioCommons API. For the final release of this tool we expect to include some more improvements particularly focused on improving pitch confidence measure, reducing single event false positives, and fixing some issues that cause processing of some files to fail.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement N° 688382

6 References

[Böck15] Böck, S., Krebs, F., & Widmer, G. (2015). Accurate Tempo Estimation Based on Recurrent Neural Networks and Resonating Comb Filters. In ISMIR (pp. 625-631).

[Brossier06] Brossier, P. M. (2006). Automatic annotation of musical audio for interactive applications (Doctoral dissertation).

[Cheveigné02] De Cheveigné, A., & Kawahara, H. (2002). YIN, a fundamental frequency estimator for speech and music. The Journal of the Acoustical Society of America, 111(4), 1917-1930.

[Degara12] Degara, N., Rúa, E. A., Pena, A., Torres-Guijarro, S., Davies, M. E., & Plumbley, M. D. (2012). Reliability-informed beat tracking of musical signals. IEEE Transactions on Audio, Speech, and Language Processing, 20(1), 290-301.

[Engel17] Engel, J., Resnick, C., Roberts, A., Dieleman, S., Eck, D., Simonyan, K., & Norouzi, M. (2017). Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders. arXiv preprint arXiv:1704.01279.

[Faraldo17] Faraldo, Á., Jordà, S., & Herrera, P. (2017, June). A Multi-Profile Method for Key Estimation in EDM. In Audio Engineering Society Conference: 2017 AES International Conference on Semantic Audio. Audio Engineering Society.

[Font16] Font, F., & Serra, X. (2016). Tempo Estimation for Music Loops and a Simple Confidence Measure. In ISMIR (pp. 269-275).

[Gómez06] Gómez, E. (2006). Tonal description of polyphonic audio for music content processing. INFORMS Journal on Computing, 18(3), 294-304.

[Gouyon06] Gouyon, F., Klapuri, A., Dixon, S., Alonso, M., Tzanetakis, G., Uhle, C., & Cano, P. (2006). An experimental comparison of audio tempo induction algorithms. IEEE Transactions on Audio, Speech, and Language Processing, 14(5), 1832-1844.

[Klapuri06] Klapuri, A. P., Eronen, A. J., & Astola, J. T. (2006). Analysis of the meter of acoustic musical signals. IEEE Transactions on Audio, Speech, and Language Processing, 14(1), 342-355.

[Knees15] Knees, P., Faraldo, A., Herrera, P., Vogl, R., Böck, S., Hörschläger, F., & Le Goff, M. (2015, October). Two Data Sets for Tempo Estimation and Key Detection in Electronic Dance Music Annotated from User Corrections. In ISMIR (pp. 364-370).

[Mauch14] Mauch, M., & Dixon, S. (2014, May). pYIN: A fundamental frequency estimator using probabilistic threshold distributions. In Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on (pp. 659-663). IEEE.

[Noland07] Noland, K., & Sandler, M. (2007, May). Signal processing parameters for tonality estimation. In Audio Engineering Society Convention 122. Audio Engineering Society.

[Percival14] Percival, G., & Tzanetakis, G. (2014). Streamlined tempo estimation based on autocorrelation and cross-correlation with pulses. IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP), 22(12), 1765-1776.

[Raffel14] Raffel, C., McFee, B., Humphrey, E. J., Salamon, J., Nieto, O., Liang, D., & Ellis, D. P. W. (2014). mir_eval: A Transparent Implementation of Common MIR Metrics. In ISMIR.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382

[Temperley99] Temperley, D. (1999). What's Key for Key? The Krumhansl-Schmuckler Key-Finding Algorithm Reconsidered. Music Perception: An Interdisciplinary Journal, 17(1), 65-100.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382