Response to reviewers comments on audio segmentation paper

Tim Scarfe, Wouter M. Koolen and Yuri Kalnishkan
Computer Learning Research Centre and Department of Computer Science,
Royal Holloway, University of London, Egham, Surrey,
TW20 0EX, United Kingdom
{tim,wouter,yura}@cs.rhul.ac.uk

October 6, 2014

Abstract

This document is a response to the reviewers comments and questions regarding the paper *Segmentation of electronic dance music*.

1 Reviewer 1

The paper is interesting, well written, and the math is correct. However, there also are several issues that must be addressed in a revision:

1. How does the method compare to existing alternatives?

   We have compared to the most popular method in the literature, which is Foote's novelty peak finding approach on a self-similarity matrix. See Section 1.1 for the literature review, Figures 1-2, and Section 7.3-8.

2. What are the details of the most closely related alternatives?

   We have expanded the literature review, please see See Section 1.1.

3. A more detailed exposition of the obtained experimental results is a must. For example, analytical confusion matrices, ROC curves, etc.

   We have added something similar to ROC curves, the F_1 scores plotted against time thresholds. It doesn't make sense to use ROC curves

because the false positive rate is the inverse of the true positive rate for this application.

We have also added several new visualizations to give some context on the results.

4. What about the computational costs of your method?

   They have been outlined in Section 4.2.* but generally speaking:

   - Discrete Fourier transform is $O(NlogN)$ where N is the next power of 2 of the number of samples in a tile.
   - Creating the Cosine matrix $O(N^2)$ where N is the number of tiles.
   - Creating the summation cost matrix is linear time $O(TWm)$
   - The *contig-static* and *contig-evolution* cost matrices modify the cosine matrix in place (linear time transformation) then use the summation routine.
   - the Symmetry cost matrix as implemented by us is currently the slowest routine, on GitHub its implemented in $O(T2W^2)$ time (see [1]) but it could also be improved to linear time.
   - the routine for calculating the posterior distribution is basically a copy of the routine for calculating the best cost segmentation but the *mins* have been swapped with *sums*, so that is also linear time.

   We ran a genetic search to optimize the parameters (see Section 7.2.2) so it is clearly fast enough to run thousands of times.

2 Reviewer 2

In this paper the authors describe an algorithm for segmenting DJ dance music streams into their respective tracks.

Although I must say I am not an expert in this field, I am myself a fan of EDM, and the problem is an interesting and complex one to try solving. It also seems that this problem has not been tackled by many researchers, so it is quite an innovative application.

[1] `https://github.com/ecsplendid/DanceMusicSegmentation/blob/master/Matlab/getcost_symmetry3.m`

What I am not so confident about is the significance: aside from EDM lovers, I do not see who else could benefit from such an algorithm. Electronic music is different from other kinds of music in the fact that tracks are made to overlap on purpose, whereas for the rest of genres as far as I know they are clearly distinguishable. Besides, the algorithm seems to be specially tailored for this particular problem, so I do not think it could be easily extensible to other 'track distinction' problem (e.g. sequences in video).

Regarding specific details, I have some questions and concerns:

1. Would it be possible for your algorithm to infer the number of tracks automatically, instead of being provided as a parameter?

   We have extended the paper to incorporate this. It is fair to say that the purpose of the paper is to do as well as possible when the number of tracks are known a priori. We do not out-perform the other methods in this configuration. However we took quite an interesting approach of using our method backwards to estimate the number of tracks. We estimate the number of tracks of many candidate number of tracks, and pick the candidate that has the least cost segmentation divided by the number of tracks. It turns out that this provides a convex function. We needed to execute a genetic algorithm to find a new set of parameters suitable for this task. Curiously; the best set of parameters for the segmentation task did not work well.

2. What is the execution time required to segment a given show? You give some hints about the computational complexity, but not about the actual time needed.

   The execution time is less than 2 seconds including all the cost matrices at the tile size of 3 seconds on a 2 hour long show. This is excluding tasks such as converting an `MP3` to `WAVE` format, down-sampling to 4KHz.

3. These two questions above are asked to assess if your method could be used in a real-life environment (e.g. Spotify, as you point out in the paper). As I see it, ideally the algorithm should be given the show to be broadcast, and it should output the track boundaries on the fly, so it should be quite fast. However, if the number and list of tracks are already known in advance, is it so important to know where they start and end?

   The algorithm is primarily aimed at the batch scenario. There are many more recordings of shows than there are live shows you can listen

to any time. The scenario you describe, when the number and list of tracks are known is precisely the scenario we want to estimate the best possible boundaries. I just went to *SoundCloud* and picked a random show (see [2]). As you can see in the top-right corner of that web page; the track listing is there – but we do not know the boundaries. This is almost always the case for all on-line radio shows. This algorithm really fills in the gap. I believe it would also be possible to adapt the algorithm for an on-line configuration but that is future work.

4. The paper would benefit greatly if you compared your algorithm with other approaches in the literature. If it is not possible to access their source codes, then you should implement from scratch at least one of your potential competitor ideas, so as to see if you outperform them or not. Comparing just with human results is useful to know that you are competitive, but it does not clarify whether you are actually the best.

   Please see Section 7.3 for this analysis. We discovered that another researcher Tom Plotz has done segmentation in the same configuration as us (see Section 1.1, 7.3.2) and compare our results to his. We are trying to get hold of his source codes for a future comparison. It is difficult to interpret his results because he gives no background on what his radio shows were or how they were annotated, and as we have demonstrated it is an error-prone process.

[2] `https://soundcloud.com/mattdarey/nocturnal-471`