Detecting Socially Significant Music Events using Temporally Noisy Labels

Yadati, Karthik; Larson, Martha; Liem, Cynthia C.S.; Hanjalic, Alan

DOI
10.1109/TMM.2018.2801719

Publication date
2018

Document Version
Final published version

Published in
IEEE Transactions on Multimedia

Detecting Socially Significant Music Events Using Temporally Noisy Labels

Karthik Yadati, Martha Larson, *Member, IEEE*, Cynthia C. S. Liem, *Member, IEEE*, and Alan Hanjalic, *Fellow, IEEE*

***Abstract*—In this paper, we focus on event detection over the timeline of a music track. Such technology is motivated by the need for innovative applications such as searching, nonlinear access, and recommendation. Event detection over the timeline requires time-code level labels in order to train machine learning models. We use *timed comments* from SoundCloud, a modern social music sharing platform, to obtain these labels. While in this way the need for tedious and time-consuming manual labeling can be reduced, the challenge is that timed comments are subject to additional temporal noise, as they occur in the temporal neighborhood of the actual events. We investigate the utility of such noisy timed comments as training labels through a case study, in which we investigate three types of events in electronic dance music (EDM): *drop*, *build*, and *break*. These *socially significant* events play a key role in an EDM track's unfolding and are popular in social media circles. These events are interesting for detection, and here we leverage the timed comments generated in the course of the online social activity around them. We propose a two-stage learning method that relies on noisy timed comments and, given a music track, marks the events on the timeline. In the experiments, we focus, in particular, on investigating to which extent noisy timed comments can replace manually acquired expert labels. The conclusions we draw during this study provide useful insights that motivate further research in the field of event detection.**

***Index Terms*—EDM, event, break, build, drop, SoundCloud, timed comments.**

I. Introduction

EVENT detection in multimedia is an important field of research and has many applications, especially with the fast growing popularity of multimedia on the web. It has been extensively studied in the context of videos, where currently a broad set of event categories at various levels of semantic complexity can be detected [1]. Research on event detection in music has, however, so far focused mainly on topics like onset detection [2], music structure segmentation [3] and auto-tagging [4]. In this paper, we look at the problem of event detection in music from a different perspective, guided by two fundamental questions:

1) What events are most interesting to detect?
2) How to detect these events effectively?

Answering these questions can be approached guided by the following consideration. A machine learning approach to event detection typically requires a large number of labels in order to train machine learning models [5]. Acquiring these labels can be an expensive and time consuming process. We can, however, benefit from the increasing contextualization of music in online social communities in order to address this problem. Users listen to music on different social music sharing platforms, such as SoundCloud or YouTube, which allow them to express their opinions/reactions to the music in the form of comments. SoundCloud, for example, offers the possibility to its users to insert *timed comments* while listening to a music track. These comments are similar to usual user comments, however, with an associated timestamp so that they refer to a particular part of the music track. Not only could such timed comments serve as training labels, reducing the need for dedicated manual annotation, but they also allow us to identify the types of events that are interesting for detection in the first place. We refer to such events as being *socially significant* as a consequence of their recognizability, popularity and anticipation. Listeners talk frequently about them in their comments. In this paper, we choose to focus on detecting these socially significant events. Examples of such events, used as a case study in this paper, are presented in Section II. For detecting these events, we choose to deploy timed comments as training labels in order to improve the training effectiveness.

Usage of timed comments as training labels, however, comes with its own challenges, in particular, the noisy nature of these comments: temporal noise. The timed comment (referring to an event) can occur precisely at the location of the actual event, in the temporal neighborhood, or far away from the location of the actual event. Fig. 1 illustrates a few possibilities of the distances between the actual event and the corresponding timed comment. Because of their noisy nature, we consider timed comments to be weak labels.

Considering the above-mentioned challenges, we propose an approach using timed comments independently as well as in combination with manually acquired expert labels to build

Manuscript received July 27, 2016; revised February 16, 2017 and May 19, 2017; accepted September 20, 2017. Date of publication February 2, 2018; date of current version August 14, 2018. This work was supported by the European Commission's 7th Framework Program under grant agreement no. 610594 (CrowdRec) and 601166 (PHENICX). The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Chengcui Zhang. *(Corresponding author: Karthik Yadati.)*

K. Yadati, C. C. S. Liem, and A. Hanjalic are with the Department of Intelligent Systems, Technische Universiteit Delft, Delft 2628 CD, The Netherlands (e-mail: n.k.yadati@tudelft.nl; c.c.s.liem@tudelft.nl; a.hanjalic@tudelft.nl).

M. Larson is with the Department of Intelligent Systems, Technische Universiteit Delft, Delft 2628 CD, The Netherlands, and also associated with Radboud Universiteit Nijmegen, Nijmegen 6525, HP, The Netherlands (e-mail: m.larson@cs.ru.nl).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2018.2801719

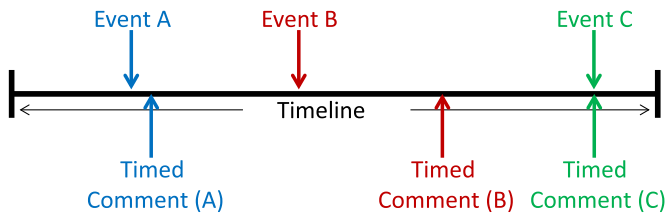

Fig. 1. Timed comments can have temporal noise. A timed comment can be in the temporal neighborhood of the actual event or precisely at the location of the actual event. Event/timed-comment pairs are in the same color.

robust machine learning models for detecting socially significant events. Specifically, we aim to answer the following research questions:

1) (RQ1) Are timed comments helpful in detecting socially significant events?
2) (RQ2) How helpful are timed comments in reducing the number of expert labels needed to train detectors?

To the best of our knowledge, our work is one of the first to use timed comments as a source of training labels for event detection in music. In this paper, we focus on the domain of electronic dance music (EDM) as a testbed for developing and evaluating our approach. This domain is interesting for investigation due to a number of socially significant event categories, as elaborated in more detail in Section II. After stating our contribution in Section III and discussing the related work in Section IV, we explain our approach and its methodological steps in Section V. We present an analysis of our dataset and evaluation metrics in Section VI. The experimental setup and results for the baseline method are described in Section VII and Section VIII presents the overall results. We then explain how the model generalizes in Section IX and evaluate our method from the perspective of a user application in Section X. Finally, we summarize our findings and provide an outlook for further research in Section XI.

II. Case-Study: Events in EDM

Electronic Dance Music (EDM) is an umbrella term for different genres of electronic music, like Techno, Dubstep, House, Electro. Producers of EDM tracks use different musical elements, like beat, tempo, sound energy or loudness, to shape the music tracks and the events occurring in them. For the purpose of this paper, we use the following set of events: *Break*, *Drop* and *Build*. They are defined as follows [6]:

- Break: A section in an EDM track with a significantly thinner texture, usually marked by the removal of the bass drum.
- Drop: A point in the EDM track, where the full bassline is re-introduced and generally follows a recognizable build section.
- Build: A section in the EDM track, where the intensity continuously increases and generally climaxes towards a drop.

These events can be considered to form the basic set of events used by the EDM producers [6]. They have a certain temporal structure internal to themselves, which can be of varying complexity. Their social significance is apparent from the presence of a large number of timed comments, related to these events, on SoundCloud. Listeners react to these events after they occur, or anticipate these events and react to them even before they occur. As an example of the latter case, the timed comment in this track[1] with the text "*Here comes the drop*" comes at the timestamp 00:50, while the actual drop happens at 01:00. While the presence of the event-related keywords in the timed comments enables us to utilize them as training labels, as it will be explained in Section V-B, their noisy distribution along the timeline, as previously mentioned, makes it an open question how useful they actually are.

III. Contribution

As reflected by our research questions in Section I, the main goal of this paper is to investigate the usefulness of timed comments as labels for training event detection models in the music audio domain. In order to provide answers to these questions, a framework is needed in which a music track is analyzed for the presence of events for which timed comments are available. There, we first identify candidate start points and then select a candidate as the predicted start point of the event using a machine learning step that is trained with noisy timed comments independently. We also combine the timed comments with expert labels. The framework uses music structure segmentation [7]. We build our framework by drawing on previous work where possible and proposing innovations where needed. The link between the previous work and the realization of our event detection framework is explained in Section IV.

The framework serves as a vehicle for obtaining insight on the helpfulness of timed comments for event detection. Our findings are communicated in the analysis and discussion of our experimental results in Sections VII and VIII. The framework design choices, such as filtering social data based on expert labels, described in Section V-B, are made in order to make it possible to answer our research questions.

In this paper, we consider the helpfulness of timed comments from two different perspectives, which correspond to two different evaluation scenarios. The first is the signal perspective and this is represented by the conventional performance metric: f-score. We analyze changes in f-score to determine whether we have improved the ability of our approach to detect and exactly localize an event. The second is a user perspective and this reflects the ability of an event detector to support user-facing applications. We choose the application of non-linear access to represent this perspective. A non-linear access system places markers for predicted events on a timeline, which allows a user to jump into the content at a particular time point. The key quantity impacting the user perception of the helpfulness of the event detection is the amount of time a user, who clicks on the marker, must wait in order to encounter an occurrence of the event. We refer to this distance as the event anticipation distance (ea_dist) and use it as an evaluation metric reflecting how users would experience the predicted start points (see Fig. 2). Section X further discusses how timed comments and very few expert labels can enable non-linear access.

[1]Link active if viewed online.

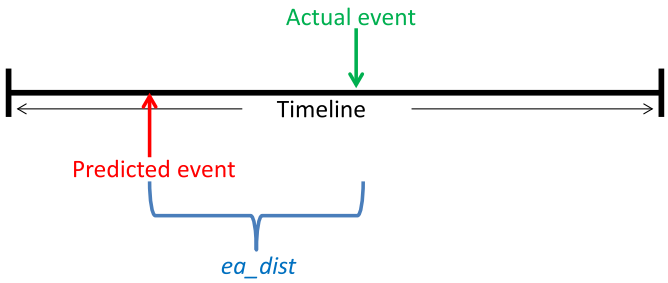

Fig. 2. Visualization of the event anticipation distance (ea_dist) metric useful to evaluate jump-in points provided to the listener in a non-linear access scenario.

IV. Related work

In this section, we provide an overview of the previous work related to our approach: audio event detection, usage of timed comments and machine learning with noisy labels. For each category, we explain to which extent we rely on the state-of-the-art, and what is new in our approach.

A. Audio Event Detection

Research related to audio event detection can broadly be divided into three categories: environmental sound recognition, music event detection and music structure analysis. Environmental sounds that can be detected in a given audio stream include, for example, bell ringing, applause, footsteps or rain. Various features and learning methods have been proposed to model the typically non-stationary characteristics of the environmental sounds [8]. We mention here as an example the usage of image processing techniques on a spectrogram image, as proposed in [9], for this purpose. These events typically come from a different acoustic source other than the background audio, while in our case, the musical events in question are part of the continuous music stream. In our paper, we use the same spectrogram image to extract features. In addition to the spectrogram image, we also explore other image representations: self-similarity matrix, auto-correlation matrix. Some other methods look specifically for the presence of speech in a given audio stream [10]. Given an audio stream, such methods also try to locate segments that contain speech and also identify attributes of speech like fricatives or non-fricatives [11], [12]. Speech related event detection in audio supports automatic speech recognition.

Event detection in music has generally focused on detecting low-level events like onsets [2]. Music onset detection is a well-studied problem in music information retrieval (MIR) and it serves as a task in the MIREX benchmark evaluation every year. Another way of approaching music event detection is music auto-tagging [4], which assigns descriptive tags to short segments of music. It is also addressed by a task in MIREX, under the name *Audio Tag Classification*,[2] where descriptive tags need to be associated with 10-second music segments. These tags generally fall into three categories: musical instruments (guitar, drums, etc.), musical genres (pop, electronic, etc.) and mood based tags (serene, intense, etc.).

[2]http://www.music-ir.org/mirex/wiki/2015:Audio_Tag_Classification

In music structure analysis [7], the objective is to divide a given piece of music into its various sections and later group them based on their acoustic similarity. It is an important task since structural elements give to a piece of music its identity. For example, in popular music tracks these structural elements could be the intro, the chorus, and the verse sections. Different aspects of musical expression have been deployed for analyzing the musical structure, such as homogeneity (e.g., in instrumentation), repeating patterns (e.g., in rhythm or melody) and novelty (e.g., through a change in tempo or tonality).

Regarding temporal analysis of the music track and event modeling using audiovisual features, in our approach we largely build on the state-of-the-art methods discussed above, as explained in more detail in Section V-C. Specifically, we deploy existing structure segmentation methods that give us an indication of the probable position of events and we use this information to distinguish between event and non-event segments. For feature extraction and event modeling, we build on spectrogram-based signal representation and on a number of proven audio features.

B. Usage of Timed Comments

Timed comments have been explored in [13] to obtain shot-level tagging of videos. In this work, a topic model is built that can link the audiovisual content of a video shot to the topic of a timed comment. The main difference with our method is that we investigate the association between the timed comments and the signal, while the authors of [13] only analyze the timed comments to achieve video shot-level tagging. A thorough investigation was conducted on timed tags used on an online video platform in [14], where the authors investigate the differences between timed and timeless tags.

YouTube allows users to mention a timestamp in a comment, which is then converted into a link to that particular part of the video. These comments are called deep–link comments and have been exploited to provide non-linear access to videos [15]. To the best of our knowledge, however, these comments have not yet been deployed for video event detection. The first attempt to do so in the music domain, which used the timed comments on the SoundCloud platform, was reported in our previous work [16] for the case study of drop event detection. The method presented in this paper, explained in detail in Section V, is an extended and improved version of the work presented in [16]. We note that it was observed in [14] that timed tags for videos are characterized by a phenomenon of temporal noise, which can considered to be comparable to the temporal noise of the timed comments in our music dataset (see Fig. 1).

C. Machine Learning With Noisy Labels

Finding effective ways of dealing with noisy labels is a critical aspect of our machine learning approach. As already mentioned, a segment containing a timed comment referring to an event might not actually coincide with the actual occurrence of that event. Consequences of this temporal noisiness of the labels could be diverse. Noisy labels could decrease classification performance, increase the complexity of the learning models

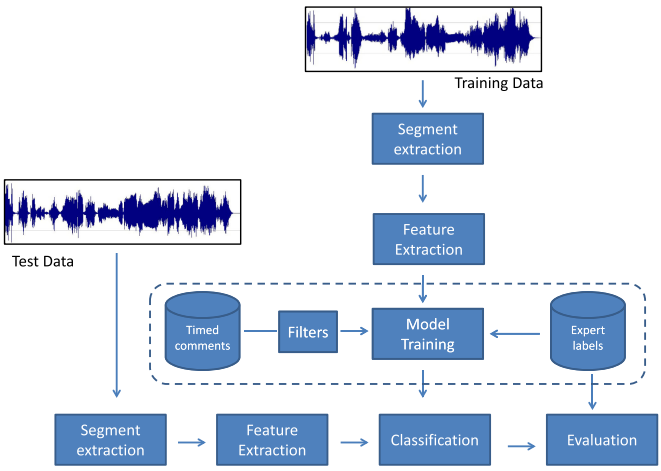

Fig. 3. A schematic view of the different steps in our approach. Note the two different sources of labels: timed comments and expert labels. Changes occur within the part of the model enclosed by the dashed line depending on the source of training labels used.

or cause difficulties in identifying relevant features. A detailed survey of different techniques to address the challenge of developing machine learning algorithms in the presence of noisy labels is provided in [17]. We address the issue of noisy labels in two ways. We use different sources of features and also propose strategies to filter the noisy labels.

V. Proposed Framework for Event Detection

We propose a machine learning algorithm that learns a model per event category, which will later be used to detect the event in a new track. We apply this algorithm to our three events of interest: drop, break and build. In addition to predicting whether an event occurs in a music segment, we also locate the start point of the event.

Fig. 3 illustrates our approach and its main methodological steps. The stage of "Filters" in the highlighted part of Fig. 3 is to filter the noisy timed comments and pass only the selected timed comments to the training stage. In the following sub-sections, we describe the different steps and explain in detail how we utilize the two different sources of labels.

A. Segment extraction

In this step, we use two different strategies used to obtain a unit of classification: Music structure segmentation (MSS) and Fixed-length segmentation (FLS). For MSS, we perform music structure segmentation on the music track and then extract fixed length classification windows centered at the segment boundaries. These windows are the unit that is used further for feature extraction, training, and prediction. The motivation behind choosing to perform structure segmentation is that the structural boundaries in a track can potentially give us the start points of different events. For example, a break is a part of an EDM track where the texture is considerably thinner compared to the rest of the track. We hypothesize that the point where the texture becomes thin will be associated with a structural boundary, and for this reason we take our unit of classification to be a window around this boundary. This hypothesis that music events occur at or near boundaries is validated later with an analysis of the dataset in Section VI-A. Exploratory experiments indicated that the music structure segmentation method proposed in [3] gives a good first approximation of the event positions in an EDM track, when compared to other segmentation methods proposed in [18] and [19]. For this reason, we use the method of [3] for MSS.

For FLS, we divide the track into fixed length segments of duration t seconds with an overlap of $t/2$ seconds between successive segments. Here, we use the full segment of t seconds as the classification unit, unlike MSS where we extract a classification window after segmentation. For this strategy, we do not have the prior knowledge provided by MSS, which means that when we use it our event detection approach becomes comparable to music auto-tagging.

B. Strategies for Deploying Training Labels

We have the timestamps of our three events of interest from two different sources: experts and timed comments (the procedure to acquire these labels is explained in detail in Section VI). Each segment coming from the segment extraction algorithm is given two labels depending on whether the timestamp given by an expert or a timed comment falls within the segment. We use four different strategies to obtain a trained model: training using expert labels (EL), training using timed comments (TC), training after combining expert labels with timed comments (CELTC) and training after combining expert labels with filtered timed comments (CELFTC). Expert labels are gold standard labels that can be relied upon and timed comments serve as weak labels. The part of Fig. 3 enclosed by the dashed line changes based on which of the above strategies we use for training.

In the EL strategy, we label a segment as a positive example for an event if an expert label falls within the segment, while the other segments are taken as negative examples. Recall that segments here refer to the classification window extracted around the structural boundary for MSS and the whole segment of t seconds for FLS. We consider this strategy (EL) to be the best possible scenario because we have labels given by experts and the model trained on these labels should be able to make a reliable prediction. We take the performance of this strategy as an upper limit and refer to the EL strategy as the baseline event detector (see Section VII-C). Other strategies (TC, CELTC and CELFTC) are deemed successful if their performance is close to the performance of the baseline event detector.

In the second strategy (TC), we label a segment as a positive example for an event if a timed comment referring to that event falls within the segment and the other segments are taken as negative examples. In the other two strategies, we divide the training data into two subsets of m and $N-m$ tracks, where N is the total number of tracks in the training set and $m = p \times N$ represents a proportion of N for $p = \{20\%, 40\%, 60\%, 80\%\}$. For example, if $p = 20\%$ then $m = 0.2 \times N$ and $N - m = 0.8 \times N$ represents a portion of the training data. We use expert labels for the m tracks and use timed comments as labels for the remaining

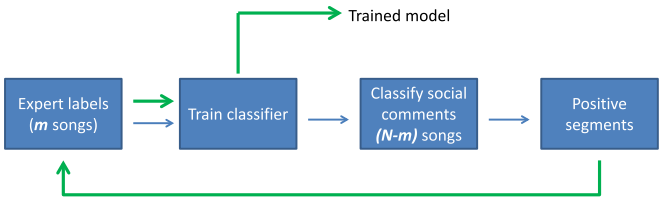

Fig. 4. CELFTC: Pipeline for combining expert labels with timed comments. This strategy involves the step of verifying the timed comments before adding them to the training data. The thicker, green arrows refer to the training after filtering the timed comments.

$N - m$ tracks. In CELTC, we directly combine expert labels for the m tracks and timed comments for the $N - m$ tracks to train a model. For CELFTC we use a different approach that includes a step of filtering the noisy timed comments (see Fig. 4). More specifically, we train a model using expert labels for m tracks and test if the timed comments from the $N - m$ tracks actually refer to the event. We then take positively classified examples from the $N - m$ tracks and add them to the existing training data labelled with expert labels i.e., m tracks. The training procedure applied to all four strategies using the corresponding sets of training labels is explained in Section V-D. In all the four proposed strategies: EL, TC, CELTC, and CELFTC, we use all the positive and negative examples for training i.e., we do not take an equal number of positive and negative examples for training.

C. Feature Extraction

The input to the feature extraction module is a fixed-length music segment (obtained from the following two strategies: MSS and FLS) and the output is a feature vector, which is then used for training a model. We explored image and audio information to choose what features to extract. Here, we provide details about the features from different sources and their corresponding dimensionality.

1) Image features: The time-frequency representation of the music signal (spectrogram) has been used in sound event recognition [20]. Fig. 5 shows the pattern representing a drop in the spectrogram. Observing Fig. 5, we can see a sweeping structure indicating the buildup followed by a sudden drop (red vertical line). We are interested in capturing such patterns, which are unique for certain events in the music. We are not looking for specific frequency values, but rather for patterns that can help us distinguish between music segments containing the event and segments not containing the event. In addition to the spectrogram, we also explore other image representations of an audio signal: auto-correlation and the self-similarity matrix, visualized as images.

In order to calculate image features, we divide each image into rectangular cells of equal size and extract second- and third-order statistical moments from these cells. We divide an image of size 738×927 into 9×9 rectangular cells of size 82×103 to compute the features. We compute the second and third order moments for all three channels: red, green and blue. Moments from cells of each channel are then concatenated to construct a feature vector with a dimensionality of 486 ($9 \times 9 \times 2 \times 3$),

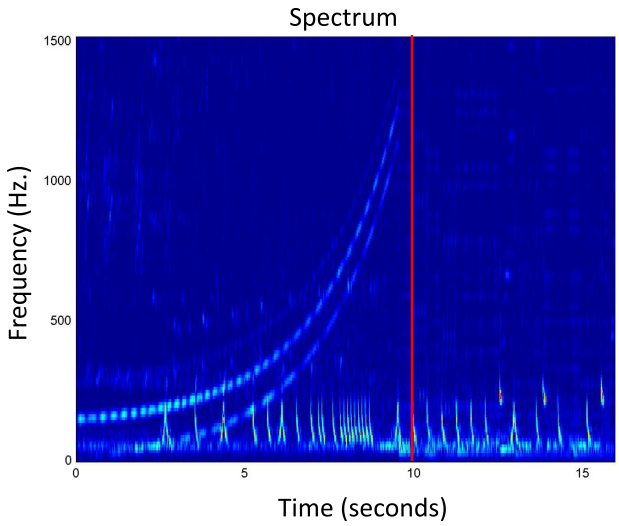

Fig. 5. Spectrogram of a segment containing a *drop*. One can observe a sweep-like structure on the left side of the figure. The red vertical line indicates the position of the drop.

which is further used to train a model. The central moment of order k (m_k) of a distribution is defined as follows: $m_k = E(x - \mu)^k$.

We use the following sets of features with the specified dimensionality: second and third central moments with rectangular cells on spectrogram (486), second and third central moments with rectangular cells on auto-correlation (486), second and third central moments with rectangular cells on self-similarity matrix from spectrogram (486), second and third central moments with rectangular cells on self-similarity matrix from auto-correlation (486).

2) Audio features: When choosing a set of audio features that will help in distinguishing a segment containing an event and a segment not containing the event, we consider the general characteristics of an audio event and focus on rhythm, timbre and dynamics as feature categories. We use the following features to capture the component of rhythm as explained in [21]: rhythm patterns (RP), rhythm histogram (RH), temporal rhythm histogram (TRH) and statistical spectrum descriptors (SSD).[3] In addition to these, we also use other features: tempo (measured in beats per minute), number of beats in a segment, average and standard deviation of the difference between the locations of successive beats.[4] In order to capture the timbral variations, we compute the statistics from the frame-wise MFCC and frame-wise zero-crossing rate (ZCR). The dynamics of the signal change over the course of the build-up towards the drop. To capture these dynamics, we use the statistics (mean, std, var, average of first order derivative, average of second order derivative) computed from the frame-wise RMS energy.

In summary, we use the following set of features with the corresponding dimensionality: RMS energy (5), MFCC (65), ZCR (5), RP (1440), RH (60), TRH (168) and SSD (420).

[3]http://www.ifs.tuwien.ac.at/mir/musicbricks/index.html#RPextract

[4]https://acousticbrainz.org/static/download/essentia-extractor-v2.1_beta2-1-ge3940c0-win-i686.zip

D. Feature Selection and Training

As observed in the previous section, the dimensionality of the features is high and this in-turn could lead to problems like overfitting or longer training times. In order to avoid such problems, we perform feature selection on the combined features from each of the two modalities (audio and image). We use a feature ranking method, where a score is computed for each dimension of the feature vector and the features are ranked based on this score. We compute the score by measuring the statistical dependency (SD) of the feature values on the corresponding class labels as done in [22]. SD is a measure that quantifies whether the feature values are dependent on the class labels or they co-occur by chance. Since we obtain a ranking of the features using this method, we need to determine which of the top-k features need to be included and we use cross-validation to make this choice.

Another important choice to make is the type of model to use. We choose a Support Vector Machine with a Radial Basis Function kernel because of its discriminative nature, simplicity and wide applicability. Here, we say a few words about why Hidden Markov Models, a common model used for time series data, are inappropriate for our problem. Hidden Markov Models work well for tasks like speech recognition and phonetic segmentation [23]. The strength of HMMs for these tasks is twofold: their ability to predict in the face of the uncertainty of event boundaries (word and phone boundaries) in the speech signal and their ability to model sequence information. In contrast, for our music event detection task, we have a high degree of certainty that an event will be located around a structural boundary. The challenge we face is uncertainty with respect to identification, rather than with respect to segmentation. In our problem, the amount of sequential information is limited to the fact that non-events alternate with events. This information is well captured by our segmentation approach, which also enforces constraints with respect to how closely two detected events can occur to each other. Although HMM architectures can be designed to capture long-distance dependencies, such designs, would come at the cost of an explosion in the number of parameters. Apriori we can anticipate such architectures to be ineffective since they ignore the constraints inherent to the structure of our problem.

With an RBF kernel, there are two parameters, which need to be optimized in an SVM: C and γ. The cost parameter C controls the trade-off between complexity of the decision rule and the frequency of error, while γ is the Gaussian kernel parameter [24]. We perform a grid-search for these parameters using cross-validation and obtain the parameters that give the best performance. We use the cross-validation data set (80% of the data) for this experiment. We carry out a nested cross-validation, which first determines the k to use for selecting the top-k features, and then determine C and γ.

1) Compute SD score for each feature dimension.
2) Pick $k = 50, 100, 150, 200, 250, 300, 350, 400$, where k indicates how many of the top-k ranked features are to be picked for training.
3) For each value of k, follow these steps:
   - Pick the top-k features.
   - Randomly split the cross-validation data into two sets: X_{train} (90%) and X_{val} (10%).
   - Take X_{train} as the new training set and perform cross-validation (grid-search for C and γ) to obtain the best performing model. Use this model to predict labels in X_{val}.
   - Repeat these steps ten times to obtain average validation performance.
4) Choose the k with the best average validation performance.
5) Select the top-k features and perform 10-fold cross-validation on the cross-validation data to obtain the best parameters: C and γ. Now train an SVM on the actual training set using these parameters, which is further used for evaluation.

This procedure is followed while training a model for the four different strategies (EL, TC, CELTC, CELFTC), as explained earlier.

E. Classification

While testing, we follow the same procedure: we first create classification units (using FLS and MSS), which yields a set of segments. We then extract features, and represent each segment using the k features that were obtained while training the model. Using the trained model, we predict labels for the segments. Since we have three events of interest: drop, break, and a build we use three binary classifiers, one for each event. The choice of having three binary classifiers, rather than a single classifier which can predict three classes of events, was made so that we can investigate the utility of timed comments as training labels for each event individually. We train models with four different strategies as explained in Section V-D, and predict labels for each test segment. For the models that use MSS, we predict the location of the event to be the mid-point of the segment, which corresponds to a structural boundary in the original segmentation. As we will see in Table II, majority of the events start at a segment boundary and hence we use the segment boundary as the start point of the event.

VI. Dataset and Analysis

Traditional music tagging datasets like MajorMiner[5] use short music clips and collect labels through crowdsourcing/gamification, while other datasets, like the million song dataset [25], consist of whole tracks and tags collected in the wild on social networks. The focus of this paper is to build a machine learning model that can localize events on the timeline and we want to achieve this goal while minimizing the labeling effort. In contrast to the existing auto-tagging datasets (mentioned above), we need data that provides time-code level labels generated by listeners through social participation. In our work, we therefore rely on SoundCloud as a source of music and the corresponding social data in the form of timed comments.

[5]http://majorminer.org/info/intro

TABLE I
EXAMPLES OF TIMED COMMENTS ON SOUNDCLOUD: TEXT AND TIMESTAMP

Timestamp	Comment
00:32	That vocal is great.. give everyone goosebump
01:01	Amazing melody
01:28	loved the drop

TABLE II
PERCENTAGE OF DIFFERENT EVENTS THAT ARE $t = 0, 1, 2, 3, 4, 5, 6$ SECONDS CLOSE TO STRUCTURE SEGMENT BOUNDARIES

Event	0 sec	1 sec	2 sec	3 sec	4 sec	5 sec	6 sec
Drop	80%	1%	0%	1%	1%	0%	1%
Build	56%	4%	6%	2%	2%	3%	10%
Break	60%	10%	5%	2%	4%	6%	2%

SoundCloud is an online social music sharing platform that allows users to upload, record and share their self-created music. Our goal is to exploit timed comments, which refer to a particular time-point in the track, and could contain useful information about the presence of events. Specific examples of comments from SoundCloud that refer to musical phenomena are given in Table I. Using timed comments on SoundCloud as a source also provides an additional advantage over independent labeling of segments: the user has more context to listen to before they react to certain parts of the music track.

We deploy the SoundCloud API[6] to collect our data. Via the search functionality we search for tracks during the year 2014 that have a Creative Commons license, which results in a list of tracks with unique identification numbers. We search the timed comments of these tracks for the keywords: *drop*, *break* and *build*. We keep the tracks whose timed comments contain a reference to these keywords and discard the other tracks.

We use the resulting 500 music tracks to evaluate our proposed method. Most commonly occurring genres in our dataset are the following: dubstep, electro and progressive house. We have a total of 640 drops, 760 builds and 550 breaks in our dataset. These numbers indicate the actual number of events in our dataset i.e., the events are counted based on the expert labels (procedure to obtain expert labels explained later in this section). Associated with the dataset , there are 720 comments with the word "drop", 750 comments with the word "build" and 600 comments with the word "break". Note that the statistics indicate the number of timed comments that have a reference to the specific events, meaning that there could be multiple timed comments for a single event posted by different users. We use the timestamps of these timed comments, containing reference to our events of interest, as training labels in the following strategies: TC, CELTF, and CELFTC.

To create the expert labels, we ask a panel of 3 experts to listen to the tracks in the dataset and mark our three events of interest on the timeline of the music track. Each expert marks the events on the timeline of a subset of the music tracks individually. In order to make sure that all the experts have a common understanding of the events and the annotation procedure, we gave them a set of 20 music tracks that are not part of this dataset, but are from the same source (SoundCloud). We ask the experts to mark the events for these 20 tracks and we find that the three experts agree on more than 90% of the annotations. After this check we then ask the experts to mark the timestamps of the events on the timeline of the music tracks. After this process, we have timestamps from two different sources: experts and timed comments, which we employ in our experiments. The dataset, containing the mp3 files, timestamps of the events (both expert labels and timed comments), is hosted on the Open Science Framework and can be accessed here: https://osf.io/eydxk/.

[6]https://developers.soundcloud.com/docs/api/guide

A. Structure Segmentation

As indicated earlier, we hypothesize that the events would happen in the vicinity of the structural boundaries. In order to validate our hypothesis, we look at the distance between the timestamps of the boundaries and the events in our training set. The training set constitutes 60% of the whole dataset and contains 411 drops, 567 builds and 345 breaks. We perform MSS on the tracks in the training set and obtain the timestamps of the boundaries. On an average, there are 13.6 segments per track in our training set.

The segment boundaries can exactly coincide with the event or can occur in the vicinity of the event. In order to have an estimate of the distance between the event and the segment boundary, we count the number of events at a fixed distance of s seconds, where $s = \{0, 1, 2, 3, 4, 5, 6\}$ and report our observations in Table II. For example, if $s = 0$ seconds then we count the number of events which coincide with the segment boundaries. Similarly, if $s = 3$ seconds we count the number of events that are 3 seconds away from a segment boundary. Examining Table II, we see that a large portion of the events ($\geq$80%) are within a distance of 6 seconds from segment boundaries. It is also interesting that 80% of the drops actually coincide with segment boundaries. These statistics support our hypothesis that the events occur within striking distance ($\leq$6 seconds) of the structural boundaries.

VII. EXPERIMENTAL SETUP AND BASELINE

In this section, we explain the experimental setup and report the results of our baseline event detector. Recall that the baseline event detector is trained on expert labels and serves as a comparison for other proposed strategies (see Section V-B). We first explain how we split our dataset for the different experiments. We then explain how we tune different parameters in our approach. As explained in Section III, we evaluate our method from two different perspectives: signal and user. This requires different evaluation metrics and we explain our choice of metrics in this section.

We split our data at the track level into three sets: 60% training data (already mentioned), 20% development data and 20% test data. We do it this way in order to ensure that we do not draw the training and testing material from the same track.

TABLE III
NUMBER OF SELECTED FEATURES AND THE TOP SELECTED FEATURES

Event	Image features	Audio features
Drop	150, Auto-correlation, Spectrogram, Similarity matrix from spectrogram	200, RP, ZCR, RMS, SSD, MFCC
Break	100, Spectrogram, Similarity matrix from spectrogram	150, MFCC, SSD, RMS, RP
Build	200, Similarity matrices from auto-correlation and spectrogram, Spectrogram	200, SSD, RP, BPM,

This split is used for most experiments. In Sections V-D and IX, cross-validation is performed on the combined training and development set (80% of the original data), which we refer to as the cross-validation set.

A. Parameters

In this sub-section, we look at how we choose values for different parameters in our method. We have two different strategies: MSS and FLS. For MSS, we first segment the track and then extract a classification window centered at the segment boundary for feature extraction. The parameter that must be set for MSS is the size of the classification window. We explore the following values: 5, 10, 15, and 20 seconds for the size of the classification window. For each value, we follow the procedure of feature selection and training as explained in Section V-D. Using this trained model, we predict the events for tracks in development set and compute the f-scores. By following this procedure, we obtain an optimal performance with 15 seconds as the size of the classification window. For FLS, we divide the track into fixed length segments of duration t seconds and use the entire segment as the classification window. We follow a similar procedure, as discussed for MSS, and obtain an optimal performance on the development data at $t = 15$ seconds.

For the audio features, we use the standard configuration provided by the tools we use for feature extraction. For the image features, we extract the spectrogram for a 15-second music segment by dividing it into 50 ms frames with no overlap. We cap the frequency at 1500 Hz, since we find a clear visible pattern for our musical events below this frequency level. Using MIRToolbox [26], we compute the spectrogram with the above-mentioned parameters and save the result as an RGB image that is further used for feature extraction. Please recall that we divide the image into 9×9 rectangular cells [9], with a cell size of 82×103 and ignore the border pixels on all 4 sides (see Section V-C1). We compute the second and third order moments from the RGB pixel values of each cell and concatenate them to obtain a single feature vector, which is further used in the classification procedure.

B. Evaluation Metrics

We use different evaluation metrics to understand various aspects of the proposed approach. As indicated earlier (see Section III), we use two different scenarios: the traditional classification and a use case (non-linear access). For the traditional classification, we use f-score for the positive (fs_+) and negative class (fs_-) as well as the average f-score (fs_{avg}). Since we are also marking the events on the timeline, we assess jump-in points by measuring the distance between start point of the actual event and the predicted event. For this we use two different distance measures: 1. Absolute distance (abs_dist), measured as the difference in timestamps of predicted position and ground-truth; 2. Event anticipation distance (ea_dist), measured as the difference in timestamps of ground truth and the most recent preceding prediction. The distance metric, ea_dist, indicates the usefulness of our method in applications like non-linear access (see Fig. 2), where the user would like to skip to the next event (see Section III). If there is no previously predicted event, ea_dist chooses the beginning of the track. However, because of the length of EDM tracks and the distribution of events, this situation does not occur in practice. The other distance metric, abs_dist, is only used for the purpose of comparison across the different strategies.

TABLE IV
F-SCORES FOR THE BASELINE EVENT DETECTOR EL: FLS USING IMAGE FEATURES

	fs_+	fs_-	fs_{avg}
Drop	70.3	96.1	83.2
Break	71.6	94.2	82.9
Build	69.8	89.9	79.8

TABLE V
F-SCORES FOR THE BASELINE EVENT DETECTOR EL: FLS USING AUDIO FEATURES

	fs_+	fs_-	fs_{avg}
Drop	68.2	92.3	80.2
Break	69.8	93.1	81.4
Build	67.9	92.4	80.1

C. Baseline Event Detector

We now report the results of our baseline event detector that uses only expert labels for the entire dataset. Tables IV and V report the f-scores: fs_+, fs_-, fs_{avg}. Similar results are also reported for MSS in tables VI and VII. Observing the scores, we can say that the features extracted from the three image representations (see Tables IV and VI) perform better than the audio features (see Tables V and VII). Of all three events, the scores for detecting the build are lower, which is understandable because it is quite difficult, even for human listeners, to locate the start point of a build.

Here, we also report the number of features that were selected for each event. Table III lists the number of features selected and

TABLE VI
F-SCORES AND DISTANCE METRICS FOR THE BASELINE EVENT DETECTOR EL: MSS USING IMAGE FEATURES

	fs_+	fs_-	fs_{avg}	abs_dist	ea_dist
Drop	73.7	97.4	85.5	2.8	2.6
Break	74.4	96.5	85.4	3.1	2.9
Build	70.2	93.1	81.6	3.4	2.9

TABLE VII
F-SCORES AND DISTANCE METRICS FOR THE BASELINE EVENT DETECTOR EL: MSS USING AUDIO FEATURES

	fs_+	fs_-	fs_{avg}	abs_dist	ea_dist
Drop	71.3	94.6	82.9	4.1	3.0
Break	71.1	95	83	4.8	3.9
Build	69.8	87.1	78.4	4.5	3.7

the top features. We observe that the rhythm-related features dominate the audio features while spectrogram and similarity matrices dominate the image features.

In addition to the f-scores, we also report two other metrics, abs_dist and ea_dist (Tables VI and VII). We report these metrics only for MSS and not for FLS, because the 15-second segments in FLS do not hold any specific meaning while the structural segments in MSS are hypothesized to be the start points of our events of interest (due to Table II). Here, it is important to note that ea_dist considers predictions that precede the actual events on the timeline i.e., the predicted start point of the event comes before the actual start point. After manual inspection, we observe that a majority of the detected events precede the actual events. We use the ea_dist metric in order to quantify how close the detection is to the actual event. The values of ea_dist and the above findings suggest that we can direct the listener to a few seconds before the actual event is heard. Further analysis and discussion on the significance of ea_dist is presented in Section X.

VIII. EXPERIMENTAL RESULTS

In this section, we report the results of the experiments that help us in addressing the two research questions as introduced in Section I. We also introduce a naive event detector that randomly picks segment boundaries as start points of our events of interest.

A. Naive Detector

In this sub-section, we describe a naive detector which picks x number of events from each tracks where x is the average number of events in the training set. In our training set, we have 1.4 drops, 1.6 builds and 1.5 breaks per track, on average. We follow these steps for the naive classifier:

- Perform MSS on each track. Recall that there are 13.6 segments, on an average, per track (see Section VI-A).
- Randomly pick x number of segment boundaries as the start points of our three events of interest, where x is as explained above for each event.
- Repeat the above step 10 times to reduce the effect of biases.
- Compute all the evaluation metrics as explained in VII-B

The performance of the naive detector is reported in Table VIII and we observe that the average f-scores are very low. We consider the performance of this naive detector as the lower bound and that of the baseline event detector (see Section VII-C) as the upper bound for comparing the proposed strategies (TC, CELTC, and CELFTC).

TABLE VIII
F-SCORES AND DISTANCE METRICS FOR THE NAIVE CLASSIFIER: RANDOMLY PICK x NUMBER OF EVENTS FROM EACH TRACK

	fs_+	fs_-	fs_{avg}	abs_dist	ea_dist
Drop	5.9	71.4	38.6	29.1	32.6
Build	4.9	61.4	37.6	28.7	33.4
Break	6.5	68.7	37.6	31.4	34.9

TABLE IX
F-SCORES FOR THE STRATEGY TC: TIMED COMMENTS AS TRAINING LABELS AND FLS USING IMAGE FEATURES

	fs_+	fs_-	fs_{avg}
Drop	29.4	60.1	44.7
Break	34.2	59.4	46.8
Build	27.9	58.6	43.2

TABLE X
F-SCORES FOR THE STRATEGY TC: TIMED COMMENTS AS TRAINING LABELS AND FLS USING AUDIO FEATURES

	fs_+	fs_-	fs_{avg}
Drop	27.2	61.5	44.3
Break	30.8	56.4	43.6
Build	29	58.4	43.7

TABLE XI
F-SCORES AND DISTANCE METRICS FOR THE STRATEGY TC: TIMED COMMENTS AS TRAINING LABELS AND USING MSS USING IMAGE FEATURES

	fs_+	fs_-	fs_{avg}	abs_dist	ea_dist
Drop	28.1	66.3	47.2	21.5	18.1
Break	33.2	52.1	42.6	24.3	21.2
Build	28.4	59.1	43.7	26.6	22.3

B. Using Timed Comments as Training Data

We now investigate the utility of timed comments as training labels, which helps us in addressing the first research question (RQ1 from Section I). We follow the same procedure as in the baseline event detector, except for the source of labels. We use timed comments instead of expert labels for training our models. Tables IX, X, XI, and XII report the results. Observing the tables, we can say that the timed comments perform very well in comparison to the naive classifier (see Table VIII), but

TABLE XII
F-SCORES AND DISTANCE METRICS FOR THE STRATEGY TC: TIMED COMMENTS AS TRAINING LABELS AND USING MSS USING AUDIO FEATURES

	fs_+	fs_-	fs_{avg}	abs_dist	ea_dist
Drop	23.1	61.2	42.2	29.4	24.6
Break	24.1	59.1	41.6	25.2	20.3
Build	31.1	56.1	43.6	31.2	29.4

not so well when compared to the baseline event detector (see Tables IV, V, VI, VII). We observe a significant improvement in fs_+, abs_dist, and ea_dist, when compared to the naive classifier. However, we see a decline in f-scores for the negative class. The classifier struggles to identify non-events, which probably have less regularity than events. We surmise that the noisy nature of timed comments makes it even harder to learn non-events. In order to ensure that the classifier is not over hypothesizing, we count the number of events that the classifier hypothesizes per track. From Section VI-A, we know that there are 13.6 segments, on average, per track in our training set. Consider the drop event detector, we use a classifier trained on timed comments alone to count the number of segment boundaries that are classified as a drop, in each track of the test set. Then we take an average of the number of drops across all the tracks in the test set. By repeating this process for the other two events, we observe that the classifier hypothesizes 3.1 drops, 3.6 builds and 2.6 breaks per track on an average. These numbers are not overly high compared to the actual average number of events per track: 1.3 drops, 1.5 builds and 1.1 breaks. In an application scenario in which the average number of events expected per track is highly stable, the prior information that is used here by our naive classifier could also be integrated into our event detection models. However, here, we will continue to assume a use scenario in which that information is not available, and not add it to our models. We can see that the timed comments are indeed useful in detecting socially significant events and thus we have an answer for RQ1. Now, we will explore the combination of timed comments and expert labels to address the next research question, where we investigate whether the presence of timed comments can reduce the number of expert labels needed to detect socially significant events.

C. Combining Expert Labels and Timed Comments

The main contribution of this paper, as presented in Section III, is the investigation of the utility of timed comments as training labels. In the previous sub-section, we saw that using timed comments alone as training labels yielded lower scores because of the noisy nature of timed comments. Here, we investigate how the addition of timed comments used as labels can reduce the number of expert labels needed for detecting socially significant events. We investigate this by performing a series of experiments focusing on the strategies: CELTC and CELFTC, introduced in Section V-B. In these strategies, we divide the training data into two subsets of m tracks and $N - m$ tracks, N being the total number of tracks in the training set and $m = p\% \times N$. We use the following values for $p = \{20\%, 40\%, 60\%, 80\%\}$, which controls the proportion of the training data (N) that is used. In CELTC, we directly combine the expert labels for the m tracks and timed comments for the $N - m$ tracks to train our model.

In CELFTC, we train a model using the expert labels on m tracks and use the model to filter the timed comments on the $N - m$ tracks. It is important to note that CELFTC requires more training time than the other strategies because it involves a two-step process of first filtering the timed comments and then re-training the model using the additional data from the filtering step. Since we use the top-k features computed in the first step of the algorithm (see Section V-D), the additional training time in the second step is not very high. For example, when $p = 60\%$, the overall training time of CELFTC is a mere 6% more than that of CELTC. After filtering the timed comments, we add the positively labelled examples from the $N - m$ tracks to the actual training set of m tracks to build the final model (illustrated in Fig. 4). For each value of m, we repeat the experiment 10 times and report the average results in order to minimize the chance of interference of incidental characteristics of the data.

In order to provide a further basis for comparison, we report the results of training with m tracks (EL@p) i.e., we use only a part of the training data with expert labels corresponding to the value of $p = 20\%$, 40%, 60%, 80%. For example, if $p = 40\%$, then we use 40% of the training data with expert labels to train the model. This model then predicts the positions of the events in the test set and we compute the f-scores as usual.

Tables XIII, XIV, XV, and XVI report the average f-scores (fs_{avg}) for each of the strategies (CELTC, CELFTC and EL@p) at different values of p. Similarly, Tables XVII and XVIII report the distance metrics for each strategy. Observing the tables, we can say that image features are more effective than audio features. Filtered timed comments (CELFTC) perform better than the unfiltered timed comments (CELTC) when combined with the expert labels. This can be observed in the results for CELFTC and CELTC, where the f-scores for CELFTC are higher than those for CELTC. When the CELFTC's performance is greater than that of EL@p, results are highlighted in bold.

Filtering the timed comments (CELFTC) seems to improve the performance beyond just using the expert labels (EL@p) at certain proportions of the training data. For example, the average f-score for detecting a drop using CELFTC, at $p = 60\%$ and $p = 80\%$, is greater than that of EL@60 and EL@80% respectively (see Table XIII). Similar observations can be made for the break at 60% and 80% of the training data. For the event build, the average f-scores of CELFTC come very close to the f-scores of EL at 80% of the training data. The distance metrics abs_dist and ea_dist reported in Tables XVII and XVIII indicate that the scores for CELFTC at 60% are very close those for EL at 60%.

Next, we further investigate the performance of CELFTC, at different proportions of expert labels, by comparing its performance with that of the baseline event detector, which represents an ideal situation. Recall that the baseline event detector was trained with expert labels on the entire training set (see Section VII-C). For the baseline event detector, we choose the following combination for all the events as it was shown to result in the best performance: MSS and Image features. For the same

TABLE XIII
AVERAGE F-SCORES FOR TRAINING USING DIFFERENT PROPORTIONS OF EXPERT LABELS FOR THE THREE DIFFERENT STRATEGIES: CELTC, CELFTC AND EL@P. RESULTS ARE FOR FLS USING IMAGE FEATURES

Event	20%			40%			60%			80%		
	CELTC	CELFTC	EL@20	CELTC	CELFTC	EL@40	CELTC	CELFTC	EL@60	CELTC	CELFTC	EL@80
Drop	43.6	45.3	50.2	56.1	61.1	64.2	65.5	**76.1**	72.4	71.6	**81**	78.1
Break	44.2	47.2	58.7	61.7	65.8	69.5	72	**80**	77.8	73.6	**82.6**	81
Build	43.2	43.8	49.3	55.8	58.7	61.3	63.7	**74.1**	73.4	71	**78.2**	77.8

TABLE XIV
AVERAGE F-SCORES FOR TRAINING USING DIFFERENT PROPORTIONS OF EXPERT LABELS FOR THE THREE DIFFERENT STRATEGIES: CELTC, CELFTC AND EL@P. RESULTS ARE FOR FLS USING AUDIO FEATURES

Event	20%			40%			60%			80%		
	CELTC	CELFTC	EL@20	CELTC	CELFTC	EL@40	CELTC	CELFTC	EL@60	CELTC	CELFTC	EL@80
Drop	44.4	45.5	49.1	53.5	56.9	58.15	66.6	**72.6**	70	75.1	**78.9**	76.3
Break	47.2	48.3	52.2	59.3	59.7	61.5	70.3	**77.8**	76.4	76.3	**80**	79
Build	43.8	43.4	46.5	54.9	57.5	59.2	65.2	73.1	74	73	76.5	76.8

TABLE XV
AVERAGE F-SCORES FOR TRAINING USING DIFFERENT PROPORTIONS OF EXPERT LABELS FOR THE THREE DIFFERENT STRATEGIES: CELTC, CELFTC AND EL@P. RESULTS ARE FOR MSS USING AUDIO FEATURES

Event	20%			40%			60%			80%		
	CELTC	CELFTC	EL@20	CELTC	CELFTC	EL@40	CELTC	CELFTC	EL@60	CELTC	CELFTC	EL@80
Drop	44.2	46.4	51	54	56.3	59.5	65	**74.7**	73	75	**81.4**	78.6
Break	52.2	52.4	54	60.1	61.6	62.5	69.9	**79.4**	78	74.6	**81.7**	79
Build	44.1	44	48.5	56.3	60.2	62.5	63.5	**72.4**	72	71.2	**77.4**	76

TABLE XVI
AVERAGE F-SCORES FOR TRAINING USING DIFFERENT PROPORTIONS OF EXPERT LABELS FOR THE THREE DIFFERENT STRATEGIES: CELTC, CELFTC AND EL@P. RESULTS ARE FOR MSS USING IMAGE FEATURES

Event	20%			40%			60%			80%		
	CELTC	CELFTC	EL@20	CELTC	CELFTC	EL@40	CELTC	CELFTC	EL@60	CELTC	CELFTC	EL@80
Drop	47.6	48	52.2	62.5	64.9	65.9	73	**75.9**	73.8	81	**83.4**	81.6
Break	49.5	50.1	53.7	69.8	72.1	72.9	78.7	**83.1**	79.7	81.3	**84**	83
Build	44.3	44.6	49.4	59.6	63.8	65.5	70.6	72.8	74.5	75.1	**81**	80.1

combination, we report the results of CELFTC and also add results for EL@p at different proportions of expert labels. The results are depicted in Figs. 6 (drop), 7 (build), and 8 (break). The blue horizontal line in the figures represents the performance of the baseline event detector (Table VI). Observing the figures, we can see that with 60% of the training data labelled with expert labels we already achieve a performance very close to the baseline event detector. For example, observing Fig. 8 at 60%, the performance of CELFTC and the performance of the baseline break event detector are almost the same. This indicates that with 60% expert labels and the addition of freely available timed comments we obtain a performance that is quite close to the performance of the baseline event detector which uses 100% expert labels. In other words, with a reduced number of expert labels (60%), we obtain a performance closer to the baseline event detector. From this result, we can conclude that if we have a training set labeled with expert labels, then, it will improve our classifier to add additional training data labeled with filtered timed comments, so long as we have a minimum amount of expert-labeled data. On this basis of this conclusion, we can say that the timed comments are helping in reducing the number of required expert labels, which represents a positive answer to RQ2.

IX. GENERALIZATION OF THE MODEL

A. Cross-Validation

A 5-fold cross-validation was performed on the cross-validation data (80% of the entire dataset) and the average

TABLE XVII
DISTANCE METRICS (abs_dist AND ea_dist) FOR TRAINING USING DIFFERENT PROPORTIONS OF EXPERT LABELS FOR THE THREE DIFFERENT STRATEGIES: CELTC, CELFTC AND EL. RESULTS ARE FOR MSS USING AUDIO FEATURES

Event	20%			40%			60%			80%		
	CELTC	CELFTC	EL@20	CELTC	CELFTC	EL@40	CELTC	CELFTC	EL@60	CELTC	CELFTC	EL@80
Drop	19.1,16.2	19.2,15.9	17.4,14.3	14,12	13.7,12	13.4,12.1	14.2,12.2	**11.5,9.6**	11.1,9.1	10.4,8.3	**8.0,6.1**	8.3,6.4
Break	17.3,15.1	17.1,15.8	15.4,14.6	13,10.2	11.7,10	11.2,10.5	11.6,9.3	**9.6,8.7**	9.4,7.9	8.6,6.8	7.3,6.4	7.4,6.1
Build	16.1,15.5	16.3,14.6	17.8,16.3	15,13.3	14.4,13.8	13.6,11.8	14.5,12.6	13.6,11.5	11.4,9.5	11.5,9.3	10.6,8.7	8.6,6.3

TABLE XVIII
DISTANCE METRICS (abs_dist AND ea_dist) FOR TRAINING USING DIFFERENT PROPORTIONS OF EXPERT LABELS FOR THE THREE DIFFERENT STRATEGIES: CELTC, CELFTC AND EL. RESULTS ARE FOR MSS USING IMAGE FEATURES

Event	20%			40%			60%			80%		
	CELTC	CELFTC	EL@20	CELTC	CELFTC	EL@40	CELTC	CELFTC	EL@60	CELTC	CELFTC	EL@80
Drop	17.4,15.3	16.8,14.9	15.3,12.4	15.6,12.9	13.7,11.1	12.4,9.9	10,8.5	**8.1,7.9**	9.5,8.6	6.3,5.1	**5.9,4.9**	6,5
Break	16.4,14.2	15.9,13.9	14.2,13.1	12.4,10.2	11.8,10.6	10.4,8.7	9,7.6	**7.4,6.9**	8.1,7.8	5.4,4.6	**4.8,3.9**	5.2,4.1
Build	18,15.2	17.8,15.6	16.4,12.8	15,12	14.2,11.9	11.4,10.6	12.4,10.6	10.8,7.1	9.6,6.8	10,8.2	**7.9,6.0**	7.6,6.4

TABLE XIX
CROSS-VALIDATION RESULTS (AVERAGE F-SCORES AND DISTANCE METRICS) FOR THE THREE EVENTS

	fs_{avg}(IM,FLS)	fs_{avg}(AU,FLS)	fs_{avg}(IM,MSS)	fs_{avg}(AU,MSS)	abs_dist(IM)	ea_dist(IM)	abs_dist(AU)	ea_dist(AU)
Drop	73.3 ($\pm$4.1)	72.2 ($\pm$3.2)	77 ($\pm$5.3)	74.4 ($\pm$4.2)	7.1 ($\pm$1.1)	5.2 ($\pm$1.2)	6.9 ($\pm$1.7)	5.4 ($\pm$3.2)
Break	73.2 ($\pm$3.1)	71.4 ($\pm$4.2)	76 ($\pm$4.1)	75.3 ($\pm$5.6)	7.2 ($\pm$2.8)	5.5 ($\pm$2.9)	7.1 ($\pm$2.1)	5.6 ($\pm$1.3)
Build	71.3 ($\pm$5.3)	72.7 ($\pm$3.6)	76.2 ($\pm$3.2)	74.4 ($\pm$5.7)	7.8 ($\pm$2.1)	5.7 ($\pm$3.0)	7.1 ($\pm$1.4)	5.8 ($\pm$4.2)

im: image features; au: audio features

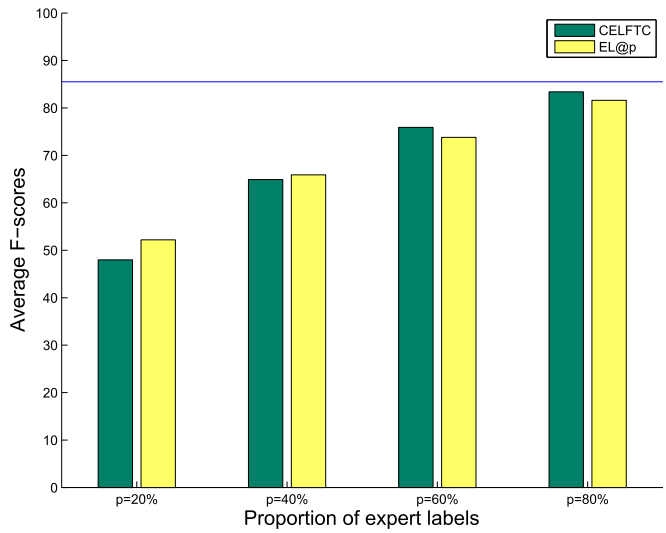

Fig. 6. Average f-scores (fs_{avg}) for detecting a drop for CELFTC: FLS and image features at different proportions of expert labels. The horizontal blue line indicates the performance of the baseline event detector with 100% expert labels.

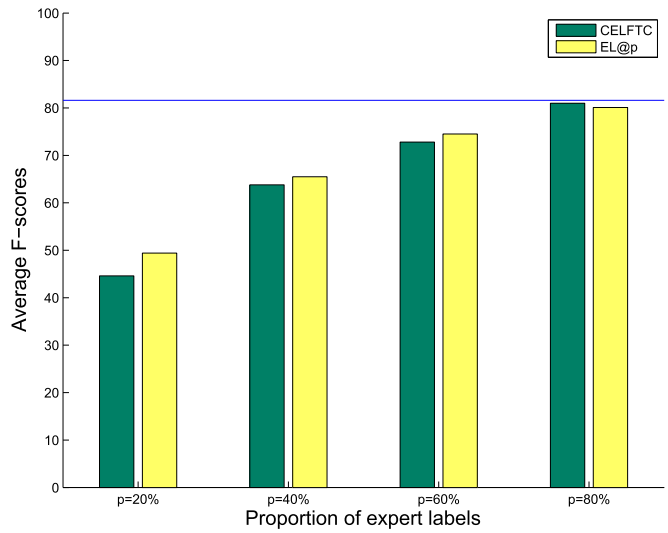

Fig. 7. Average f-scores (fs_{avg}) for detecting a build for CELFTC: MSS and audio features at different proportions of expert labels. The horizontal blue line indicates the performance of the baseline event detector with 100% expert labels.

f-scores and standard deviation are reported in Table XIX. One of the reasons to perform a cross-validation experiment is that the dataset is relatively small and we want to investigate whether the trained model overfits. Results of the cross-validation are good but lower when compared to the ones reported in Tables IV, V, VI, and VII.

This effect can be related to our sampling method. For the purpose of cross-validation, the folds are created at the track

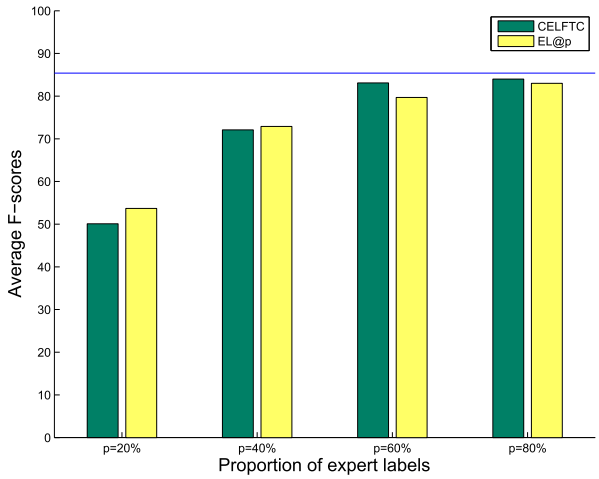

Fig. 8. Average f-scores (fs_{avg}) for detecting a break for CELFTC: MSS and image features at different proportions of expert labels. The horizontal blue line indicates the performance of the baseline event detector with 100% expert labels.

TABLE XX
AVERAGE F-SCORES FOR CELFTC ON DATA FROM A NEW SOURCE (YOUTUBE) FOR DIFFERENT PROPORTIONS OF EXPERT LABELS

Event	F-score for 60% expert labels	F-score for 100% expert labels
Drop	73.2	76.4
Break	74.9	77.1
Build	71.4	73.5

level, and not at the event level. This is necessary in order to ensure that it is never the case that training and testing material is drawn from the same track. However, the track-level sampling makes the folds sensitive to the presence of one or two tracks with a style of event that is overall more "difficult" (applies in particular to short events). For this reason, the variance between the folds is higher than expected and the average is lower. The lower average raises a question on the generalization capability of the model and in order to answer this question, we turn to another dataset. Specifically, we next report the results of the experiment on an unseen dataset that provide an insight into the generalizability of the model.

B. Performance on Data From a New Source

In order to check for the generalizability of the model, we conduct another experiment where we take the test set from another source. YouTube contains many EDM tracks and can be used as another source of music data. We download 70 tracks from YouTube and manually marked the positions of our three events in the tracks. We use this as the test set and the corresponding ground-truth in order to evaluate the performance of the detector. We chose our best model in order to predict the events on the new test set. We use MSS and image features for evaluation. We use two different trained models that use 60% and 100% expert labels respectively. Table XX presents the results of the event detection on the YouTube test set. Please note that we use the same model trained for CELFTC at 60% expert labels (see Section VIII-C) and EL with 100% expert labels (see Section VII-C) for the two columns in Table XX.

Observing the scores, we can see that the performance of the event detector is reasonable and similar trends can be found when compared to the performance on the test set from SoundCloud. For example, the f-scores for both 60% and 100% expert labels are very close together.

X. EVALUATION WITH USER-PERSPECTIVE METRICS

In this section, we turn to a deeper discussion of the implication of our results for a real-world application. Specifically, we consider a non-linear access system, i.e., a system that would allow a listener to browse through the events in a track. Such a system would involve a play bar in which music events are marked, making it possible for listeners to listen specifically to certain events, without having to listen to the track entirely. For example, such a system would be useful to a DJ who is interested in quickly reviewing all the drops in a particular EDM track.

In order to understand the usefulness of our music event detection approach to users of a non-linear access system, we make use of the metric event anticipation distance, *ea_dist*, introduced in Section II, where it is illustrated in Fig. 2. Recall, that *ea_dist* is the time that a listener would need to wait before jumping into a music stream, and hearing the event that is marked on the play bar. For comparison, we also discuss the absolute distance, *abs_dist*. Note that we do not consider *abs_dist* to be a user-perspective metric, since it has the same value whether the listener is dropped into the stream before or after the event. A music event that occurs *before* a user jumps into a stream will be missed, and can, for this reason, be considered useless in a non-linear access application scenario.

When we consider this application scenario, and *ea_dist*, the full potential of timed comments becomes clear in a way not directly reflected by the f-score that has been the focus of the previous sections. We would like to draw attention to the condition in which the music event detector is trained only with timed comments as training labels and in which MSS with image features is used. This condition was presented in Table XI (see Section VIII-B). From Table XI we see that using timed comments only, we can provide a jump-in point, on an average, 18.1 seconds before the actual drop. We point out that an error of 18.1 seconds may not be substantial enough to impact user experience significantly. Statistics calculated on our dataset as a whole reveals that a typical build-drop combination can last somewhere between 6 and 20 seconds. If we can direct the user to 18.1 seconds before the drop, there is a good chance that the build will have already started and it will be obvious to listeners that they are moving towards the drop.

In the rest of this section, we make some other observations about our results from the perspective of our distance-based evaluation metrics *abs_dist* and *ea_dist*. These results are reported in Tables XI and XII (training on timed comments only) and Tables XVII and XVIII (mixing expert labels and timed comments.) Note that in Tables XVII and XVIII results are given in the order *abs_dist*, *ea_dist*, separated by a comma. Overall,

the image features are more effective than the audio features. This observation is consistent with the observations that we have made using the average f-score in previous sections. Further, we note that *ea_dist* is systematically smaller than *abs_dist*. This observation is interesting, since it means that our approach to music event detection tends to detect an event before it occurs, rather than after it occurs. In other words, it shows a tendency away from the sort of error that would be most detrimental to the user experience.

Finally, we make another observation about Tables XVII and XVIII. We see that in general, if expert labels are available, it is most advisable to train with expert labels. Adding examples labeled with timed comments to the expert-labeled training data can add another performance boost, or at least will not hurt the performance substantially. It is interesting to consider the implications of the performance that can be achieved with a relatively limited number of expert labels. For example, using 60% expert labels we see that ea_dist for the build reaches a value of 8.6 seconds for image features (see Table XVIII). This value is very close to the minimum length of a build-drop combination, again as estimated by statistics calculated on our dataset as a whole. This example suggests that listeners might not notice further improvement of *ea_dist*. It also suggests that careful attention should be paid to whether further improvements of *ea_dist* actually hurt the user experience by cutting off context that users need to fully recognize and appreciate certain music events.

XI. Conclusion and Outlook

This paper has demonstrated the utility of timed comments as a source of labels to train models to detect socially significant music events. Through experiments, we show how timed comments can be utilized as training labels independently as well as in combination with expert labels. The important conclusions of our paper are summarized here:

- Timed comments, on their own, can be used as training labels to detect socially significant events. They perform reasonably well in applications like non-linear access, where the listener wants to jump through different events in the music track without listening to it in its entirety.
- Adding expert labels improves the performance. Our experiments demonstrate that with a combination of 60% expert labels and 40% timed comments, we can potentially obtain a performance very close to the performance when we have 100% expert labels for training data.
- The performance of the event detection is not dependent on the source of data, as we obtain a good performance on an unseen test set, from YouTube, by using a model trained on SoundCloud data.

In this paper, we have presented an extended case-study on using timed comments to detect events in the music signal. Our work is one of the first to utilize timed comments as training labels to develop an event detector. We hope that our results would encourage researchers to explore the usefulness of timed comments for other media. We do not claim that the exact segment-based approach that we take here will transfer directly to videos. However, we would like to point out that our work has demonstrated that the impact of temporal noise can be overcome and that the contribution of timed comments to video event detection is worth investigating further.

References

[1] Y.-G. Jiang, S. Bhattacharya, S.-F. Chang, and M. Shah, "High-level event recognition in unconstrained videos," *Int. J. Multimedia Inf. Retrieval*, vol. 2, no. 2, pp. 73–101, 2013.

[2] J. Schluter and S. Bock, "Improved musical onset detection with convolutional neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2014, pp. 6979–6983.

[3] J. Serrà, M. Müller, P. Grosche, and J. L. Arcos, "Unsupervised music structure annotation by time series structure features and segment similarity," *IEEE Trans. Multimedia*, vol. 16, no. 5, pp. 1229–1240, Aug. 2014.

[4] H.-Y. Lo, J.-C. Wang, H.-M. Wang, and S.-D. Lin, "Cost-sensitive multilabel learning for audio tag annotation and retrieval," *IEEE Trans. Multimedia*, vol. 13, no. 3, pp. 518–529, Jun. 2011.

[5] Z. Fu, G. Lu, K. M. Ting, and D. Zhang, "A survey of audio-based music classification and annotation," *IEEE Trans. Multimedia*, vol. 13, no. 2, pp. 303–319, Apr. 2011.

[6] M. Butler, *Unlocking the Groove: Rhythm, Meter, and Musical Design in Electronic Dance Music*, (Profiles in Popular Music). Bloomington, IN, USA: Indiana Univ. Press, 2006.

[7] M. Müller, *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*. New York, NY, USA: Springer-Verlag, 2015, pp. 167–236.

[8] S. Chachada and C. C. J. Kuo, "Environmental sound recognition: A survey," in *Proc. Asia-Pac. Signal Inf. Process. Assoc. Ann. Summit Conf.*, 2013, pp 1–9.

[9] J. W. Dennis, "Sound event recognition in unstructured environments using spectrogram image processing," Ph.D. dissertation, Nanyang Technological University, Singapore, 2014.

[10] A. Brutti, M. Ravanelli, P. Svaizer, and M. Omologo, "A speech event detection and localization task for multiroom environments," in *Proc. Joint Workshop Hands-Free Speech Commun. Microphone Arrays*, 2014, pp. 157–161.

[11] J. Li and C.-H. Lee, "On designing and evaluating speech event detectors," in *Proc. Interspeech*, 2005, pp. 566–569.

[12] S. Ziegler, B. Ludusan, and G. Gravier, "Towards a new speech event detection approach for landmark-based speech recognition," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2012, pp. 342–347.

[13] B. Wu, E. Zhong, B. Tan, A. Horner, and Q. Yang, "Crowdsourced time-sync video tagging using temporal and personalized topic modeling," in *Proc. ACM Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 721–730.

[14] P. Xu and M. Larson, "Users tagging visual moments: Timed tags in social video," in *Proc. Int. ACM Workshop Crowdsourcing Multimedia*, 2014, pp. 57–62.

[15] R. Vliegendhart, M. Larson, B. Loni, and A. Hanjalic, "Exploiting the deep-link commentsphere to support non-linear video access," *IEEE Trans. Multimedia*, vol. 17, no. 8, pp. 1372–1384, Aug. 2015.

[16] K. Yadati, M. Larson, C. C. S. Liem, and A. Hanjalic, "Detecting drops in electronic dance music: Content based approaches to a socially significant music event," in *Proc. Int. Soc. Music Inf. Retrieval Conf.*, 2014, pp. 143–148.

[17] B. Frénay and M. Verleysen, "Classification in the presence of label noise: A survey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 5, pp. 845–869, May 2014.

[18] L. Barrington, A. Chan, and G. Lanckriet, "Modeling music as a dynamic texture," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 18, no. 3, pp. 602–612, Mar. 2010.

[19] R. J. Weiss and J. P. Bello, "Identifying repeated patterns in music using sparse convolutive non-negative matrix factorization," in *Proc. Int. Soc. Music Inf. Retrieval Conf.*, 2010, pp. 123–128.

[20] J. Dennis, H. D. Tran, and H. Li, "Spectrogram image feature for sound event classification in mismatched conditions," *IEEE Signal Process. Lett.*, vol. 18, no. 2, pp. 130–133, Feb. 2011.

[21] T. Lidy and A. Rauber, "Evaluation of feature extractors and psycho-acoustic transformations for music genre classification," in *Proc. Int. Soc Music Inf. Retrieval Conf.*, 2005, pp. 34–41.

[22] J. Pohjalainen, O. Räsänen, and S. Kadioglu, "Feature selection methods and their combinations in high-dimensional classification of speaker likability, intelligibility and personality traits," *Comput. Speech Language*, vol. 29, pp. 145–171, Nov. 2013.

[23] D. T. Toledano, L. A. H. Gomez, and L. V. Grande, "Automatic phonetic segmentation," *IEEE Trans. Speech Audio Process.*, vol. 11, no. 6, pp. 617–625, Nov. 2003.
[24] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.
[25] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *Proc. Int. Soc. Music Inf. Retrieval Conf.*, 2011, pp. 591–596.
[26] O. Lartillot and P. Toiviainen, "MIR in Matlab (II): A Toolbox for musical feature extraction from audio," in *Proc. Int. Soc. fMusic Inf. Retrieval Conf.*, 2007, pp. 127–130.

Karthik Yadati received the Master of Science degree from the National University of Singapore, Singapore, in 2010. He is working toward the Ph.D. degree at the Multimedia Computing Group, Delft University of Technology, Delft, The Netherlands. He worked as a Software Engineer with the IBM India Software Labs (2003–2007) before pursuing the Master of Science. His research interests include social multimedia analytics, music information retrieval, and affective computing. He regularly reviews for conferences and journals in the areas of multimedia and music information retrieval: ACM MM, ISMIR, IEEE TMM.

Martha Larson works in the area of multimedia retrieval and recommender systems. She is a Professor with Radboud University, Nijmegen, The Netherlands in the area of multimedia information technology, and is also affiliated with the multimedia computing group, Delft University of Technology, Delft, The Netherlands. Previously, she researched and lectured in the area of audio-visual retrieval at Fraunhofer IAIS and at the University of Amsterdam. She is the cofounder of the MediaEval Benchmarking Initiative for Multimedia Evaluation. She has served on the program committees of numerous conferences in the area of information retrieval, multimedia, recommender systems, and speech technology. From 2013 to 2016, she was the scientific coordinator of CrowdRec, a European project on combining crowdsourcing and recommender systems. She is currently an Associate Editor for the IEEE TRANSACTIONS ON MULTIMEDIA. Recently, she has been the Area Chair at ACM Multimedia, TPC Chair at ACM ICMR, and Tutorial Chair at ACM RecSys. In 2012, she was an Innovation Chair at IEEE ICME and organized a "Time Machine" session on the ongoing impact of early innovations. In 2016, she was the Chair for Brave New Ideas at ACM Multimedia, with the theme "Societal Impact of Multimedia Research."

Cynthia C. S. Liem (M'16) received B.Sc., M.Sc., Ph.D. degrees in computer science from Delft University of Technology, Delft, The Netherlands, and the B.Mus., M.Mus. degrees in classical piano performance from the Royal Conservatoire of The Hague, Hague, Netherlands. She is an Assistant Professor with the Multimedia Computing Group, Delft University of Technology. Her research interests include music and multimedia search and recommendation, and increasingly shift towards making people discover new interests and content which would not trivially be retrieved. Beyond her academic activities, she gained industrial experience at Bell Labs Netherlands, Philips Research, and Google. She initiated and co-coordinated the European research project PHENICX (2013–2016), focusing on technological enrichment of symphonic concert recordings with partners such as the Royal Concertgebouw Orchestra. She was the recipient of the Lucent Global Science and Google Anita Borg Europe Memorial scholarships, the Google European Doctoral Fellowship 2010 in Multimedia, and a finalist of the New Scientist Science Talent Award 2016 for young scientists committed to public outreach. As a musician, she still has an active performing career, particularly with the Magma Duo (with Emmy Storms, violin), which has been award-winning, both nationally and internationally.

Alan Hanjalic (M'99–SM'08–F'16) is a Professor and the Head of the Multimedia Computing Group with the Delft University of Technology, Delft, The Netherlands. His research interests include multimedia search, recommender systems, and social media analytics. He was the Chair of the Steering Committee of the IEEE TRANSACTIONS ON MULTIMEDIA and an Associate Editor-in-Chief for the IEEE MULTIMEDIA MAGAZINE. He was a Keynote Speaker at the IEEE International Workshop on Multimedia Signal Processing 2013, the International Multimedia Modeling Conference 2012, and the Pacific-Rim Conference on Multimedia 2007. He has been a member for the Editorial Board of several scientific journals in the multimedia field, including the IEEE TRANSACTIONS ON MULTIMEDIA, the IEEE TRANSACTIONS ON AFFECTIVE COMPUTING, *ACM Transactions on Multimedia Computing, Communications, and Applications*, and the *International Journal of Multimedia Information Retrieval*. He was General Chair and Program Chair for the organizing committees of major multimedia conferences, such as ACM Multimedia, ACM CIVR/ICMR and IEEE ICME.