CLASSIFYING DERIVATIVE WORKS WITH SEARCH, TEXT, AUDIO AND VIDEO FEATURES

Jordan B. L. Smith, Masahiro Hamasaki, Masataka Goto

National Institute of Advanced Industrial Science and Technology (AIST), Japan
{jordan.smith, masahiro.hamasaki, m.goto}@aist.go.jp

ABSTRACT

Users of video-sharing sites often search for derivative works of music, such as live versions, covers, and remixes. Audio and video content are both important for retrieval: "karaoke" specifies audio content (instrumental version) and video content (animated lyrics). Although YouTube's text search is fairly reliable, many search results do not match the exact query. We introduce an algorithm to classify YouTube videos by category of derivative work. Based on a standard pipeline for video-based genre classification, it combines search, text, and video features with a novel set of audio features derived from audio fingerprints. A baseline approach is outperformed by the search and text features alone, and combining these with video and audio features performs best of all, reducing the audio content error rate from 25% to 15%.

Index Terms— content classification, derivative works, fingerprinting, multimedia retrieval

1. INTRODUCTION

Online video-sharing websites like YouTube and Nico Nico Douga thrive on the millions of derivative works that users post to them: popular original pieces often inspire hundreds of covers, remixes, dance routines, and so forth. In Nico Nico Douga's VOCALOID community, uploaders add semantic links to the content they reuse, which permits networks of derivative works to be mapped easily [1]. On YouTube, however, derivative works must be discovered via text search, which can be imprecise. For example, in a search for "Madonna, Like a Prayer, dance routine", fewer than half of the top 20 results (as of this writing in April 2017) are videos of dances; of the others, most are dance remixes of the song. Despite interest in developing richer browsing options for users [2], YouTube's search filters remain limited to basic properties like upload date and duration.

We propose a system, outlined in Figure 1, to scan YouTube and find and classify derivative works. The system can recover the semantic links that were never created and be used to improve music video search options for users.

This work was supported in part by JST ACCEL Grant Number JPMJAC1602, Japan.

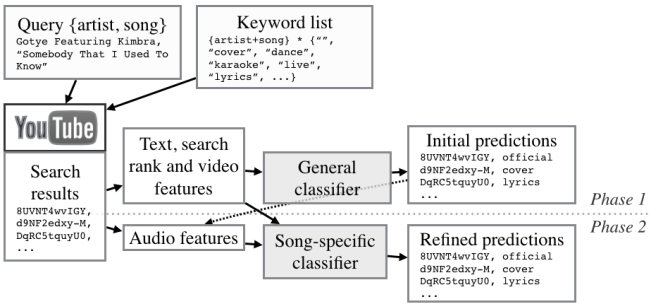

Fig. 1. Overview of two-step classification algorithm.

Classifying derivative works is complex because many independent parts of a video's audio and video content may be re-used, re-performed, or replaced with new content [1]. The score or audio content may be reused to make a cover or remix; a video of a dance routine may use the original song edited for length; a karaoke backing track is another derivative, so an amateur performance at a karaoke bar may be a second-order derivative. Often, the line between the original and the derivative is hard to decide. For example, many cover videos depict a musician re-performing one part of a song while the original recording plays in the background.

We present a novel system for discovering and classifying derivative works on a video-sharing website based on multiple features. Our main contributions are: (1) the definition of a novel classification problem; (2) the design of the audio fingerprint-based features; and (3) the evaluation of our method on a new collection of ground truth.

2. RELATED WORK

Many problems in Music Information Retrieval (MIR) can be understood broadly as "music version recognition" tasks. These include: the recognition of identical recordings, or fingerprinting [3]; the recognition of cover songs, derivatives performed by other artists [4]; the recognition of remixes [5] or instances of sampling [6]; and the recognition of live performances by a given artist [7].

The main challenge of fingerprinting is to find a compact

characterization of audio that is robust to additive noise and distortion, but also highly specific to a recording's identity. Shazam's landmark fingerprint algorithm uses pairs of nearby peaks in the audio spectrogram to generate hashes for a song; if even a few hashes from two songs collide *and* have the same time offset, this reliably indicates a match [3]. This algorithm was re-purposed to detect sampling [6], since it works with short audio queries (under 10 seconds), and is robust to addition (e.g., added musical layers) and subtraction (e.g., re-use of only the vocals). However, it is not designed to recognize time-stretched or pitch-shifted audio; a four-peak landmark was proposed to overcome this limitation and used to recognize tracks in DJ mixes [8]. While spectral landmarks are apt for recognizing identical recordings, detecting a re-performance requires a more flexible similarity function. Rather than hashing small sets of landmarks, [7] proposed learning audio filters that optimally discriminate different songs for a given artist.

Lying at the farthest end of the dissimilarity spectrum, cover songs by other artists may differ significantly from the original in terms of instrumentation, key, timbre, etc. Cover detection methods mostly aim to characterize the melody or chord structure, which are usually preserved in a cover, and then to compare recordings with sequence-matching techniques like dynamic time warping [4].

These music version recognition tasks have each been evaluated in isolation, but there has been little study of how to discern the subtle distinctions between different types of derivative works. The combination of the tasks is more complex than it may appear: consider (1) a remix; (2) a live rendition by the original performer; (3) a video taken in a dance studio with an abridged version of the original recording played on a boombox and re-captured diegetically. To a cover song detection algorithm, a live version detection algorithm, and a remix detection algorithm, all of these would be detected as derivative works, but all are distinct types of derivatives.

To solve this new problem of distinguishing between them, we use an approach similar to [6], but extend the audio features in a novel way. We consider the narrow categorization task, instead of the retrieval scenario typical in fingerprinting, in which a large database (representing "all music") is scanned for relevant items. Our approach resembles that of [7] for recognizing live works: they assumed the artist was known, essentially solving a 'song-classification' task.

As a categorization problem, our work is informed by the extensive literature on categorizing audio (see [9] for a review) and video (see [10] for a typical example). Audio classification—by genre, mood, composer or style—usually aims to identify musical features which are consistent between classes, such as tonal, timbral, or other spectral profiles. Building an effective classifier is often just a matter of devising an appropriate set of features and then training a support vector machine (SVM), or any other model, to learn the difference between the classes. Video classification works the same way, usually using features that characterize colour, texture, and movement; simple feature sets and SVMs often solve typical classification tasks with error rates under 10%, as noted by [11]. Extra information may come from text and audio features and second-order features extracted by, e.g., face-recognition [10] or speech-recognition systems [11]. Although our proposed system uses a novel audio comparison feature, our pipeline resembles previous video classification systems.

Although video classification problems often include 'music' as a class alongside 'news', 'sports', and other broadly different genres, music video classification has rarely been investigated. The authors of [12] classifed music videos by genre, while [13] classified music videos by mood. We are not aware of previous work that categorizes music videos according to their type of derivative work.

Our research thus diverges from previous work in three ways: first, our system works with an existing retrieval system, i.e., YouTube, so nearly all of the content we must classify is known *a priori* to be related to the original work in some way. Second, using YouTube as a source means we can leverage video, text and search data in addition to audio content. And third, our system must be able to understand narrow differences in types of derivative works.

3. PROPOSED SYSTEM

We introduce a music video discovery system that finds and classifies derivative works. The system depends on YouTube's search API[1], which provides search rank and basic video information, including title. Our system also requires access to each video's audio and video content, which can be downloaded from YouTube. The input to the system is a song query (including the title and performing artist) and a list of keywords (see Figure 1). The system repeatedly submits the query along with different keywords to YouTube. For each video, we compute search rank, text, video and audio features. Classification happens in two stages: first, videos are classified by a general model trained with search rank, text and video features. Second, the confidently-labeled examples are used to compute audio features that characterize how closely a video's audio content resembles that of videos belonging to each category, and these song-specific features, along with the previous features, are used to train a new classifier.

The quantity of music videos on YouTube is enormous, so our choice of artist/song queries and search keywords is arbitrary. We queried YouTube with the Billboard end-of-year Hot 100 for 2012[2], using 12 keywords: choreo, concert, cover, dance, guitar, karaoke, live, lyrics, piano, remix, tutorial, and [*empty*], i.e., the artist and song name alone. The

[1] https://developers.google.com/youtube/v3/

[2] http://www.billboard.com/charts/year-end/2012/hot-100-songs

Audio class	**Definition**
official	original audio by recording artist
cover	performance by different artist
instrumental	version without vocals
live	live performance by original artist
remix	re-arrangement of original stems
tutorial	instructions on how to perform song
Video class	**Definition**
official	official video by original artist
dance	contains dancing (possibly animated)
karaoke	contains lyrics animated to indicate timing
live	shows any artist performing
lyrics	contains printed, static lyrics
slides	series of static images
still	single static image
tutorial	instructional video
other	none of the above

Table 1. Definitions of audio and video content labels.

query set included a variety of genres and artists, and since 2012 was a few years ago, plenty of derivative works were ready to be discovered. The keyword set also ensured a variety of derivative works.

Annotating the true audio content (AC) and video content (VC) of the results is time-consuming and subjective, and we could only annotate a subset of the results: these 100 songs and 12 keywords led to 160,382 unique search results containing at least the title of the song. The first author annotated a set of roughly 600 videos, collected from 10 of these queries.[3] The videos were randomly selected from among those with peak search rank (in any keyword) of 1, 20, 100, and 300, to test accuracy at specific search depths. The AC and VC labels used are listed and briefly explained in Table 1.

4. FEATURE DESIGN

4.1. Search rank

A video's search rank under the various keywords relates to the AC and VC it is likely to contain. For video i, we take the vector of search ranks $R_i = \{r_{i1}, r_{i2}, \ldots, r_{i12}\}$, corresponding to the video's ranks for the 12 keywords. When video i is not in the list of search results for the j^{th} keyword, we set $r_{ij} = 501$, i.e., one more than the maximum search depth used. We then take the log of this feature vector.

4.2. Text features

Another clear way to guess the content of a video is to look for keywords in its title. We use Latent Dirichlet Allocation (LDA) [14] to estimate topics from titles. LDA assumes that each document in a corpus is related to one or a few topics, and that each word in it is generated by one of its topics. Estimating LDA topics and topic probabilities for each document, and using supervised learning to link topics to real categories, is a common approach to text classification. On short texts, LDA has been known to give poor results, due to the sparsity of word co-occurences (e.g., [15]). Video titles are short, but their vocabulary is highly constrained, so LDA is an apt tool.

To ensure LDA learns general topics, we only consider words that appear in at least M unique video titles, and among at least N unique song/artist queries. We remove from the title all stopwords like 'the' and 'and', and the name of the artist and song. We allow fuzzy matches to avoid learning common misspellings of artist names as important topic words. If no artist name is detected, we add "noartistname" to the title; LDA treats titles as bags of words so this is not disruptive.

We also optionally add a "knowledge-boosting" (KB) step by detecting words from vocabularies that we anticipated would relate to video content: lists of genres, instruments, names, and locations combed from Wikipedia and other sources, including the Natural Language Toolkit (NLTK) word collections [16]. We also used the parsedatetime[4] Python package to detect dates in the titles. If one of these words is detected, we append a flag word like "[name]" or "[genre]" to the title.

Topic	Top 5 words
0	[name] dance noartistname video music
1	[name] noartistname cover remix vs
2	[instrument] cover piano guitar [name]
3	[hasLyrics] lyrics karaoke hd video
4	[genre] [name] blanktitle cover parody
5	[place] live [date] [name] tour
6	[name] [place] noartistname cover live
7	[name] remix noartistname cover espanol
8	remix [name] [genre] dj dubstep
9	cover [name] acoustic live [date]

Table 2. Example topics learned from subset of Billboard Top 100 with "knowledge-boosting" applied.

Table 2 shows example topics learned from over 140,000 video titles generated by 90 song queries drawn from the 2012 Billboard Top 100, with $M = 100$ video titles, $N = 5$ queries, and with KB applied. As an example, topic 5 shows that the correlation between the words 'tour', 'live', and places and dates was learned.

4.3. Video features

The set of video categories of interest (see Table 1) is limited, and the contrast between the classes is stark: e.g., amateur

[3] Annotation data and algorithm output can be found here: `https://github.com/jblsmith/icme2017`.

[4] `https://github.com/bear/parsedatetime`

covers are often filmed with a fixed camera and little motion; remixes are often accompanied by a still image; lyrics videos are usually slideshows; etc. We are not aiming for video fingerprinting or scene recognition to, say, identify dancing.

We thus use a standard set of low-level video descriptors. Following [17], these include: (1) lighting key, a measure of overall brightness; (2) a one-dimensional estimate of the variance in colour; (3) the magnitude of the frame-to-frame change in colour, also used to detect cuts; (4,5) the magnitude and direction of the optical flow, for detecting motion; and (6) a binary flag for whether the motion content indicates multiple objects moving in contrary motion or not.

To identify lyrics and karaoke videos, we also used the general-purpose text recognition package Tesseract[5] to detect and transcribe text in each frame. Text recognition can be simple when the location of the text is known, and successful karaoke lyric transcription systems have been based on this [18]. We lack these contraints and expect many errors, so we calculate the percentage of transcribed words that appear in NLTK's 'words' corpus. The text recognition step adds two features: (7) a binary flag to indicate if any text was found; and (8) the percentage of these words that occur in NLTK's corpus of existing words.

To minimize computation time, we downloaded the lowest-quality versions of the videos available (usually 256x144 pixels) and undersampled videos to two adjacent frames per 20 seconds. For features 3–6 above, we computed the mean at both timescales (i.e., for single frame steps and for 20-second steps); the others were averaged over the video.

4.4. Audio features

We model the similarity between two audio files by extending Van Balen's method [6], itself a re-purposing of Dan Ellis' `audfprint`[6] implementation of the Shazam algorithm [3]. We use [6]'s parameter recommendations: query song q is divided into 12-second chunks with 50% overlap, and landmarks are computed at 36 per second, maximum 10 pairs per peak, peak spreading of 30. Each chunk q_i (containing n_{q_i} landmarks) could match another song p in multiple places; we consider only the closest match (with $m_{q_i,p}$ hashes), if one exists. The distance between q_i and p is characterized by the number of shared hashes: $d(q_i, p) = 1/(1 + m_{q_i,p})$.

The above approach is used in [6] to detect sampling, but this metric alone may not be sufficient to distinguish all types of derivates, some examples of which are illustrated in Table 3). Moreover, we want to distinguish cases where a few chunks match exactly, vs. where many chunks match weakly. We thus calculate the following metrics: (1) average distance of all chunks: $d(q, p) = \frac{1}{N}\sum_1^N d(q_i, p)$; (2) average distance of matching chunks: $d(q, p) = \text{mean}(d(q_i, p)|d(q_i, p) < 1)$; (3) the converse distance $d(p, q)$, since d is not a symmetric measure; (4) $|q \cap p|/|q|$, the fraction of segments in q that match any segment in p, and its converse $|p \cap q|/|p|$; (5) mean length of sequences of parallel matches; (6) if p is the original song, the transposition (-2 -1, 0, +1 or +2 semitones) that best matches q.

To compute metric 6, we add transposed versions of the official song (which is reliably detectable using search rank) to the fingerprint database. Metrics 4 and 5 require a fixed distance threshold for two chunks to count as a match. We consider two thresholds: $H \geq 1$ and $H \geq 6$ matching hashes.

Let us call this audio feature vector $AF(q, p)$. At minimum, we will compute this for the song p that is most likely to be the official audio, based on our initial predictions in Phase 1. However, we can also compare q to the top-t videos most likely to contain the official audio, and keep the result with the highest similarity. We test with $t = 1$ or $t =$ all predicted songs. Moreover, we can add extra feature vectors $AF(q, p_{cat})$ for the other content types to characterize how similar q is to to the other AC types (covers, remixes, etc.).

Audio content	**Possible example**
Original audio	$q_1 \mid q_2 \mid q_3 \mid q_4 \mid q_5 \mid q_6 \mid q_7$
Live performance	$q_1 \mid q_2 \mid - \mid q_4 \mid - \mid q_6 \mid q_7$
Remix (reuse of sample)	$- \mid q_3 \mid - \mid q_3 \mid - \mid q_3 \mid q_3 \mid -$
Tutorial	$- \mid - \mid q_1 \mid q_2 \mid q_3 \mid q_4 \mid - \mid - \mid - \mid -$
Cover	$- \mid - \mid - \mid - \mid - \mid -$

Table 3. Schematic examples of derivative works. Note that live performances by the original artist may feature pre-recorded backing tracks.

5. IMPLEMENTATION AND EVALUATION

We queried YouTube with the Billboard Top 100 for 2012 and annotated a subset of 600 results related to 10 different songs. Of these, audio and video data were able to be retrieved for 562. Since the LDA analysis is unsupervised, we trained the topic model on all the videos for the 90 songs in the Top 100 not in the annotation set: 142,377 video titles altogether, totalling just over 335,000 words (with stopwords and artist and song names removed), 1,038 of them unique.

YouTube search results provided a baseline: for each video, we assign AC and VC based on the keyword that returned that video with the highest rank. For example, videos which ranked highest under the "remix" search were assigned AC "remix" and VC "still". We generously construct the mapping post-hoc, finding the optimal many-to-one mapping of keyword to AC and VC.

We first tested the use of each feature individually, and then tested the impact of adding the features together one by one. Classification was always performed by an SVM with a linear kernel and $C = 1$. The average accuracy for audio

[5] https://github.com/tesseract-ocr/tesseract

[6] https://github.com/dpwe/audfprint

Feats	AC acc.	AC std.	VC acc.	VC std.
S	0.699	0.052	**0.705**	0.065
T	**0.781**	0.036	0.690	0.054
V	0.416	0.040	0.505	0.057
A	0.623	0.109	0.552	0.048
ST	0.822	0.021	0.767	0.055
STV	0.815	0.023	**0.804**	0.041
STVA	**0.847**	0.014	0.781	0.020
YT	0.746	-	0.685	-

Table 4. Mean AC and VC classification accuracy and standard deviation for proposed combinations of feature sets (**S**earch, **T**ext, **V**ideo, **A**udio) as well as the **YT** baseline.

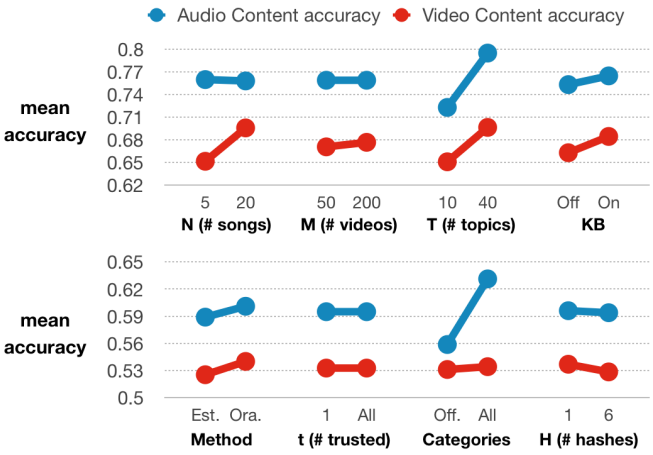

Fig. 2. Main effect plots for impact of text (top) and audio (bottom) parameters on AC and VC classification accuracy.

and video content classification, estimated with 5-fold cross validation, is shown in Table 4 along with the baseline results. In each fold, 8 songs were allocated to the train set and 2 to the test set. The number of feature dimensions varied among the SVMs (from a low of 12 for the search feature, to a high of 112 for *STVA*). Since our system design is complex and the evaluation data are limited, text and audio feature parameters were chosen after a factorial evaluation using the same data. (Text: $\{N = 20, M = 50, T = 40$, KB on$\}$; audio: $\{t = 1$, all categories, $H = 1\}$.) Although this means the final results could be inflated by overfitting, the comparisons between models and baseline are still sensible.

There were four independent settings to vary when generating the text features: the minimum number of video titles M and minimum number of song/artist queries N in which a word had to appear to be used in the LDA model, the number of topics T to estimate, and whether to include the detected vocabulary terms ("knowledge-boosting"). We conducted a 2^4 factorial experiment with $M \in \{50, 200\}$, $N \in \{5, 20\}$, $T \in \{10, 40\}$, and KB off or on. The main effects (see Figure 2) show that larger M and N (leading to smaller dictionaries) was helpful for estimating VC; on the other hand, having more numerous, and hence more specific topics, was always better. The knowledge-boosting step was also helpful.

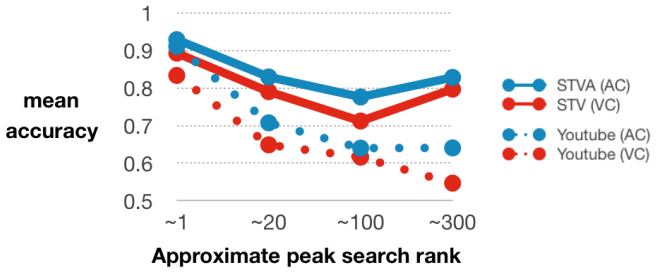

Fig. 3. Accuracy as a function of approximate search rank of videos in YouTube search.

The audio features had another four factors to vary, giving another 2^4 factorial design. The audio features are computed not on isolated songs (as with, say, MFCCs) but by comparing each song to other tentatively-labeled songs; these labels are either estimated using an SVM trained on the search, text and video features (using the text parameters just recommended), or taken from the ground truth ('oracle'). The oracle method provided a boost, as expected (see Figure 2). The number of audio comparisons made within each category, t, had little impact, but including audio comparisons with all the AC categories instead of only the official category helped greatly. Surprisingly, setting a modest threshold of similarity to avoid spurious matches (at least 6 hashes) did not improve results.

Finally, we compared combinations of features. The proposed system uses search, text and video features in a first pass, then refines its estimates with audio features. The results seem to justify this scheme, except that video features are less useful for predicting AC, and audio features less useful for predicting VC. For AC, text alone exceeds the baseline (T: 0.78); adding search features improves it (ST: 0.82); and all features together gives the best performance (*STVA*: 0.85). For VC, search alone exceeds the baseline (S: 0.71); adding text improves it (ST: 0.77), and adding video features gives the best performance (STV: 0.80). Based on this result we should amend the system design: for classifying VC, the second phase is unnecessary.

Figure 3 shows that the classification accuracy of the baseline drops significantly for later searches, for both AC and VC. In contrast, our predictions are more robust to search rank. That said, the high accuracy at greater search depths could be a result of the skewed distribution of content types (later results were likelier to be covers).

Investigating typical errors made by the system, we found that many AC estimation errors arose from confusion between the 'cover' and 'live' labels. In hindsight, these labels conflate two independent factors: authorship (original vs. cover artist) and performance setting (live vs. studio). Reconsidering our approach, we recommend replacing the 6 broad cat-

egories of AC with a larger set of binary attributes in future work. For example, our 'live' label could be split into flags for 'vocal track is re-performed', 'vocal performer is artist', and 'recorded in live setting'. VC estimation errors were more evenly distributed, though the system notably did not learn the difference between 'still' and 'slide' videos, possibly a result of our choice to undersample, and the fact that many videos labled 'still' had introductory animations.

6. CONCLUSION AND FUTURE WORK

We have studied a previously uninvestigated task: the discovery and classification of derivative musical works. Our proposed method is based on an existing content retrieval system, YouTube, but by combining multiple sources—search ranks, titles, video and audio content—it appears to classify works better than YouTube's search tools alone seem able to provide. However, our evaluation was limited to fewer than 600 items, so the absolute accuracy of the system cannot be guaranteed. The system could improve a content search interface by allowing users to filter by derivative type, or to ensure a variety of content types in general searches.

Our system uses YouTube's search API, but the framework is applicable to other video sharing sites. Different LDA topic models would be required for different sites, since the models are sensitive to conventions for titling videos on a given service. Future work could also incorporate information available on YouTube but not taken advantage of here, such as knowledge of the type of content in other videos uploaded to a given channel.

7. REFERENCES

[1] M. Hamasaki and M. Goto, "Songrium: A music browsing assistance service based on visualization of massive open collaboration within music content creation community," in *Proc. Int. Symp. on Open Collaboration*, NY, NY, 2013, pp. 4:1–10, ACM.

[2] K. Tsukuda and M. Goto, "ExploratoryVideoSearch: A music video search system based on coordinate terms and diversification," in *Proc. IEEE Int. Symp. on Multimedia*, 2015, pp. 221–224.

[3] A. L.-C. Wang, "An industrial-strength audio search algorithm," in *Proc. ISMIR*, Baltimore, MD, 2003.

[4] J. Serrà, E. Gómez, and P. Herrera, *Advances in Music Information Retrieval*, chapter Audio Cover Song Identification and Similarity: Background, Approaches, Evaluation, and Beyond, pp. 307–332, Springer, Berlin, Heidelberg, 2010.

[5] M. Casey and M. Slaney, "Fast recognition of remixed music audio," in *Proc. IEEE ICASSP*, 2007, vol. 4, pp. 1425–1428.

[6] J. Van Balen, "Automatic recognition of samples in musical audio," M.S. thesis, Universitat Pompeu Fabra, Barcelona, Spain, 2011.

[7] TJ Tsai, T. Prätzlich, and M. Müller, "Known-artist live song ID: A hashprint approach," in *Proc. ISMIR*, 2016, pp. 427–433.

[8] R. Sonnleitner, A. Arzt, and G. Widmer, "Landmark-based audio fingerprinting for DJ mix monitoring," in *Proc. ISMIR*, 2016, pp. 185–191.

[9] 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, 2011.

[10] H. K. Ekenel, T. Semela, and R. Stiefelhagen, "Content-based video genre classification using multiple cues," in *Proc. Int. Work. on Automated Information Extraction in Media Production*, NY, NY, 2010, pp. 21–26, ACM.

[11] M. Rouvier, S. Oger, G. Linarès, D. Matrouf, B. Merialdo, and Y. Li, "Audio-based video genre identification," *IEEE/ACM Trans. on Audio, Speech, and Language Process.*, vol. 23, no. 6, pp. 1031–1041, 2015.

[12] A. Schindler and A. Rauber, *Advances in Information Retrieval: Proc. Eur. Conf. on IR Research*, chapter An Audio-Visual Approach to Music Genre Classification through Affective Color Features, pp. 61–67, Springer International Publishing, Cham, Switzerland, 2015.

[13] A. Yazdani, E. Skodras, N. Fakotakis, and T. Ebrahimi, "Multimedia content analysis for emotional characterization of music video clips," *EURASIP J. on Image and Video Process.*, vol. 2013, no. 1, pp. 1–10, 2013.

[14] David M. Blei, Andrew Y. Ng, and Michael I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, 2003.

[15] L. Hong and B. D. Davison, "Empirical study of topic modeling in Twitter," in *Proc. Workshop on Social Media Analytics*, NY, NY, 2010, pp. 80–88, ACM.

[16] S. Bird, E. Loper, and E. Klein, *Natural Language Processing with Python*, O'Reilly Media Inc., Sebastopol, CA, 2009.

[17] Z. Rasheed, Y. Sheikh, and M. Shah, "On the use of computable features for film classification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 1, pp. 52–64, 2005.

[18] Y. Zhu, K. Chen, and Q. Sun, "Multimodal content-based structure analysis of karaoke music," in *Proc. ACM Multimedia*, Singapore, 2005, pp. 638–47, ACM.