COMPUTATIONAL STRATEGIES FOR BREAKBEAT CLASSIFICATION AND RESEQUENCING IN HARDCORE, JUNGLE AND DRUM & BASS

Jason A. Hockman,
DMT Lab,
Birmingham City University
Birmingham, United Kingdom
jason.hockman@bcu.ac.uk

*Matthew E. P. Davies**
Sound and Music Computing Group,
INESC TEC
Porto, Portugal
mdavies@inesctec.pt

ABSTRACT

The dance music genres of hardcore, jungle and drum & bass (HJDB) emerged in the United Kingdom during the early 1990s as a result of affordable consumer sampling technology and the popularity of rave music and culture. A key attribute of these genres is their usage of fast-paced drums known as breakbeats. Automated analysis of breakbeat usage in HJDB would allow for novel digital audio effects and musicological investigation of the genres. An obstacle in this regard is the automated identification of breakbeats used in HJDB music. This paper compares three strategies for breakbeat detection: (1) a generalised frame-based music classification scheme; (2) a specialised system that segments drums from the audio signal and labels them with an SVM classifier; (3) an alternative specialised approach using a deep network classifier. The results of our evaluations demonstrate the superiority of the specialised approaches, and highlight the need for style-specific workflows in the determination of particular musical attributes in idiosyncratic genres. We then leverage the output of the breakbeat classification system to produce an automated breakbeat sequence reconstruction, ultimately recreating the HJDB percussion arrangement.

1. INTRODUCTION

1.1. Background

During the early 1990s, DJ-oriented electronic musicians in the United Kingdom embraced affordable sampling technologies (e.g., Akai S950), allowing them to replicate and manipulate recorded sounds without the need for reverse engineering or synthesis. These technologies, and the innovative techniques developed to harness these technologies resulted in the development of three new genres: hardcore, jungle and drum & bass (HJDB). A unique attribute of these genres is their integration of short segments of percussion solos from funk and jazz recordings known as breakbeats.

While melody and harmony are often-used attributes in the determination of an artist's ability within many genres, perhaps the most revealing characteristic by which an HJDB producer, track (herein used to describe a complete musical piece) or subgenre might be individuated, is through breakbeat selection and creative usage of breakbeats. Breakbeat selection is the choice of one or more breakbeats taken from the vast amount of existing funk and jazz recordings. Breakbeats are typically recorded into a sampler's memory, and an arrangement is created by reordering MIDI notes associated with segments of the breakbeat within a sequencer—also termed *resequencing*. Breakbeat usage in HJDB centres on the creative transformation of an initial source breakbeat through any number of techniques available to HJDB producers. These transformations may include pitch-shifting, time-stretching, filtering, resampling, and resequencing. If the undertaken process is viewed as a pipeline with the original source breakbeat as the audio input and the transformed breakbeat as the output, we can consider a breakbeat transformation to be a kind of complex audio effect. HJDB producers (e.g., Bay B Kane, Justice, DJ Krust) are well known for their choice of source breakbeat material and recognisable for the types of transformations they employ. More recently, artists such as Fracture and Om Unit have led a resurgence in the popularity of using breakbeats, and continue to redefine the aesthetic of breakbeat manipulation. Hardcore, jungle and drum & bass exist as genres within a continuum of musical influence that involves the cultural and generational borrowing of stylistic cues and sonic artefacts (i.e., breakbeat samples) [1], similar to North America's hip hop, which also relies on the history and sound palette of funk and jazz.

1.2. Motivation

Automated analysis of breakbeat usage is of interest both from a musicological perspective—to gain insight on a technology-driven composition process—as well as in terms of how to recreate the stylistic idiom of HJDB producers. Analysis of rhythmic modification of breakbeats can provide insight into the resequencing practices undertaken by HJDB producers in individual tracks, which could, in turn, be used to assess these practices across an artist's career or across subgenres. Automating this procedure would be useful to music producers in search of novel drum sequences for use in their productions, or as an educative recommendation tool for musicians to learn how others have structured their own percussion arrangements.

In pursuit of both these goals, we explore techniques for the automatic recognition of breakbeats as a critical first step towards understanding how they have been used and repurposed within HJDB. Identification of source samples has become a popular online activity, and the ability to "spot" samples is considered a mark of a good sample-based musician. Whosampled is an online community of musicians and avid listeners that collectively attempts to document information related to sample usage.[1] The site allows users to explore how sampled music has been appropriated

* MD is financed by National Funds through the FCT - Fundação para a Ciência e a Tecnologia within post-doctoral grant SFRH/BPD/88722/2012.

[1] www.whosampled.com

by other musicians and to provide links between new and old material.

At present there are only a handful of tools that have been presented that seek to automate the sample identification process. Balen et al. [2] adapted the audio fingerprinting method of Wang [3] to the task of sample identification. Here the *spectral peak* or *landmark* method is used to identify a series of salient peaks in the spectrogram that are similar to those in other recordings. While the spectral peak method has been shown to be robust to noise and distortion, it is less useful for the detection of percussion, or in the context of heavily transposed, or *pitched* material [2], which is often the case with breakbeats. Alternatively, Dittmar presents a sample plagiarism tool designed to assess similarity between musical excerpts by evaluating the correlation between the time-varying gains representing basis activations as generated through non-negative matrix factorisation [4]. Whitney adapted Dittmar's model to assess similarity using Pearson's correlation [5].

In this paper we investigate two classification scenarios for determining the presence of breakbeats in music that has incorporated them. First, we attempt to identify tracks that use a given breakbeat within a dataset with a limited set of known labels to determine the general validity of the approach under controlled conditions. Second, we attempt a more realistic formalisation as a binary classification problem directed towards the presence or absence of a given breakbeat within a database of HJDB music with a much larger number of breakbeats. Following this classification stage, we then automatically extract rhythmic and metrical information in the form of onset and downbeat locations to facilitate the segmentation of breakbeats and hence the eventual reverse engineering to relate the output (i.e., transformed breakbeat) back to the source input.

We consider the presented techniques to be part of computational research in DJ-oriented electronic music, much of which has been pioneered by Collins [6, 7, 8], who developed the *bbcut* real-time breakbeat segmentation and resequencing tools intended to replicate the idiomatic breakbeat manipulations of the genre.

The remainder of this paper is structured as follows: Section 2 outlines our breakbeat classification method. Section 3 presents our evaluations and datasets. Section 4 presents our proposed method for breakbeat sequence reconstruction and reproduction that utilises the breakbeat classification performed in Section 2. Finally, Section 5 provides conclusions and areas for future work.

2. METHOD

Figure 1 depicts an overview of the components in our breakbeat resequencing analysis. At the core of the system is the proposed breakbeat classification method depicted as black boxes. The white boxes show additional processing stages required for breakbeat rearrangement (Section 4).

HJDB producers select breakbeats for their unique rhythms and timbres [9], which are the result of a percussionist's performance on a drum set captured in a particular room using a specific set of recording devices (e.g., microphone, preamplifier, mixing board). The difference between drum hits of different breakbeats can therefore be relatively small, however for a sample identification system to be useful for the task of finding breakbeats it must be capable of identifying heavily manipulated and relatively short audio queries within large collections [2]. The system should also be able to identify the individual segments of a breakbeat (e.g., an individual bass drum hit), as breakbeats are very often resequenced,

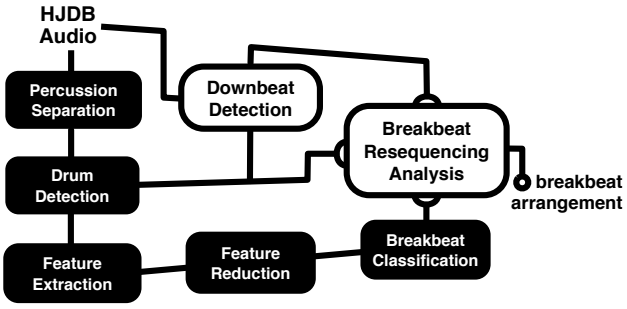

Figure 1: *Overview of proposed breakbeat resequencing analysis method with specialised processing. HJDB audio enters into stages of the breakbeat classification system (black boxes). Additional processing stages (white boxes) are undertaken to achieve the breakbeat rearrangement.*

or may only appear as brief segments—that is, not in their entirety. We therefore propose a specialised solution, which extracts and classifies individual drums from the musical timeline using a combined signal processing and machine learning approach. Our formalisation is motivated by the actual breakbeat usage in HJDB music, in which segments of multiple breakbeats may be used within the same track.

2.1. Multi-breakbeat Classification

The first breakbeat classification method attempts to select the underlying breakbeat from a set of known breakbeat classes in a multi-class classification problem; for example, to determine if the breakbeat in an HJDB track is of *Amen*,[2] *Funky Mule*,[3] or *Apache* [4] origin. We apply this design as a simplified problem to test the validity of the proposed model, however it is also a necessary formalisation for solving ties in the event that more than one breakbeat has been selected in a series of harder binary classification problems (e.g., *Amen* versus *non-Amen*, where *non-Amen* is comprised of examples with any number of breakbeats present).

Within the above context, we first concentrate our efforts on the detection and classification of bass drums as they are the drum class within the standard drum kit that tend to exhibit the least amount of timbral overlap with other sounds. Snare drums in comparison tend to exhibit more overlap with other sounds as they extend over a wider range of frequencies. The presented breakbeat classification method therefore seeks to take advantage of the uniqueness of salient bass drum timbres in each breakbeat. Our proposed method is shown in Figure 1 as the collection of black boxes. To identify regions containing bass drums, HJDB audio (mono .wav files sampled at 44.1 kHz with 16-bit resolution) is entered into a two-stage process of harmonic suppression and drum detection. We incorporate the median-filtering approach of FitzGerald [10] for harmonic suppression (window size = 4096 samples, hop size = 1024 samples) to reduce the contribution of the non-percussive instruments prior to drum detection. To perform drum detection, we create a prototypical spectral profile of a bass drum d as the average spectrum of multiple breakbeat bass

[2] The Winstons – *Amen, Brother* (1969)
[3] Ike Turner & The Kings of Rhythm – *Funky Mule* (1969)
[4] Michael Viner's Incredible Bongo Band – *Apache* (1973)

drums. As shown in Eq. (1) we cross-correlate the harmonically-suppressed spectrogram $S(m, k)$ (with frames m and bins k) with bass drum spectral profile $d(k)$ to produce a bass-drum frequency spectrogram $B(m, k)$.

$$B(m, k) = S(m, k)d(k) \tag{1}$$

Following the standard spectral difference approach for generating an onset detection function (e.g., [11]) we then create a bass-drum detection function by measuring spectral difference in the bass-drum frequency spectrogram, B, similar to the approach presented by Davies et al. [12]:

$$\Gamma_{BD}(m) = \sum_{k} H(|B(m, k)| - |B(m-1, k)|) \tag{2}$$

where $H(x)$ is the half-wave rectifier operation. Peak-picking Γ_{BD} provides a set of temporal indices used as the initial frames for feature extraction in the bass-frequency spectrogram features (BFS). These features are extracted starting at each detected bass drum position and are six time frames in length (approximately 210 msec) and represent frequencies between approximately 50 Hz to 325 Hz (bins 4–30). In addition to features extracted from the BFS, 13 MFCCs are extracted from the same bass drum regions as found by the bass-drum detection function. In the final step of the feature extraction stage, the means and standard deviations of each row and column of the BFS and MFCC features are extracted (time and frequency independently) from each bass drum event to create feature matrix J (100 features x # bass drum events).

Prior to training a model we first obtain reduced dimensionality feature matrix J' by applying principal component analysis (PCA) to J, reducing the number of features to t principal components, where t represents the smallest value that retains 95% variance. To perform the classification we use the support vector machine (SVM) algorithm, which is trained using feature matrix J', and an associated class vector C that contains the breakbeat class names associated with each bass drum. The SVM is implemented using the LIBSVM C-SVC algorithm with an RBF kernel [13]. The γ and c parameters for the SVM were established ($\gamma = 2^{-3}$, $c = 2^{0}$) using grid search with three-fold cross-validation using ten tracks in each class. To perform classification of test audio A, feature matrix J_A is created in a similar fashion to feature matrix J. J_A is projected onto the PCA coefficients from the training stage resulting in the reduced set of features J'_A. A classification decision is made for each extracted bass drum in J'_A, resulting in multiple classifications for each test example. An overall class is then determined using majority voting. Ties are resolved using additional classification stages performed with only those classes that are tied.

2.2. Binary Classification

The method presented in 2.1 is capable of determining the presence of an arbitrary breakbeat if provided sufficient training material for each of the breakbeats under assessment. However, collecting a sufficient amount of ground truth for a large number of breakbeats is a difficult task that requires expert listeners. To approach the real problem of identifying an underlying breakbeat within a set of music that contains an unknown, large number of breakbeats, we alter the model in two ways: first, we reduce the number of possible class labels to two, and second, we remove the majority voting component needed to break ties. The latter alteration also affords the more grounded scenario in which multiple breakbeats may be present in one example track.

In addition to classification by SVM, we investigate the use of a deep network classifier in the provision of binary class labels with the aim of increasing classification accuracy through learning additional relationships in the data. Deep network classification involves the transformation of input features into output labels through a series of layered interconnected networks that produce transitional representations. The classifier is constructed in the Theano Python library for deep learning ([14, 15]) and uses same initial set of features J as constructed in 2.1. The network contains a single hidden layer (with 256 nodes) and is trained through minibatch stochastic gradient descent (learning rate = 0.02, epochs = 50000) with batch-wise loss assessed using mean negative log-likelihood.

3. EVALUATION

In order to assess the appropriateness of our selected problem formalisation and the suitability of the methods applied to breakbeat classification, we perform two evaluations based on the multi-class and binary classification systems outlined in Section 2.

3.1. Evaluation 1: Multi-class Breakbeat Classification

The first evaluation is used to determine the validity of a classification based on subtle timbral variations inherent between breakbeats, and to test the worth of the specialised processing stages (i.e., drum segmentation, specialised feature set). In this experiment we consider the simplified problem of a multi-class classification with three breakbeat classes—*Amen*, *Apache*, and *Funky Mule*. These breakbeats were chosen based on their prominence in the HJDB music catalog.[5]

The evaluation is conducted using a dataset comprised of examples of HJDB music, each containing one of the three breakbeats. Annotations for breakbeats were provided by expert listeners (HJDB producers and DJs) through queries in separate threads on the Facebook social media platform, in which HJDB musicians were asked to list their favourite HJDB tracks that use one of the three breakbeats. Musicians listed candidate tracks, and often confirmed or rejected candidate tracks provided by others. In addition, HJDB tracks with the specified breakbeats that were mentioned in interviews by musicians were also added as candidate tracks [9]. All excerpts were originally in .wav or .mp3 format ($\geq$192 kbps) and between 15 seconds and two minutes in length. In total there were 93 excerpts (31 per class).

We test three configurations of the system presented in 2.1. The first configuration is trained using only the MFCC-based features (`M-SVM`); the second configuration is trained using only BFS-based features (`B-SVM`); the third configuration is trained using both BFS-based and MFCC-based features (`BM-SVM`). In addition to these specialised models, we include a fourth, generalised music classification model (`M-GMM`), as it is our hypothesis that a generalised music classification system would not be capable of differentiation between the subtle timbres exhibited between tracks containing different breakbeats. Whereas the `M-SVM` uses an SVM to classify the temporal indices of detected bass drum events, the `M-GMM` uses Gaussian mixture models to learn parameters from MFCCs extracted across frames of audio [16].

[5] e.g., www.whosampled.com/The-Winstons/Amen,-Brother/

	M-GMM	M-SVM	B-SVM	BM-SVM
Amen	74.2%	80.6%	61.3%	**83.9**%
Apache	74.2%	67.7%	77.4%	**90.3**%
Funky Mule	41.9%	54.8%	**93.5**%	87.1%
Avg.	63.4%	67.7%	77.4%	**87.1**%

Table 1: *Accuracies of breakbeat classification systems* (M-GMM, M-SVM, B-SVM *and* BM-SVM) *for multi-class classification using HJDB examples containing only Amen, Apache, and Funky Mule breakbeats along with cumulative mean accuracies (Avg.). Bold scores denote the best scores in each breakbeat class and average score.*

The four methods are evaluated using leave-one-out cross validation. For the three versions of the specialised system, class membership was determined by majority voting for the number of drum events g extracted (g = 7). The selection of the events was based on the amplitude of the peaks in the bass-drum detection function (see Section 2.1). The M-GMM assigns breakbeat class membership to the set of extracted features with the smallest negative log-likelihood from trained class models.

Table 1 summarises the results of the tested methods (M-GMM, M-SVM, B-SVM, and BM-SVM) using the HJDB breakbeat dataset. Accuracies are provided in percentages for each breakbeat class, and total accuracies are calculated as the mean across classes for each system. The generalised audio classification system, M-GMM, performed reasonably well in classifying tracks from the *Amen* and *Apache* breakbeat categories (74.2% accuracy in each class); however, performance was drastically lower for the *Funky Mule* breakbeat class (41.9%). The M-SVM system achieved slightly better results than the M-GMM system for the *Amen* and *Funky Mule* breakbeat classes, with a slight reduction in performance in the *Apache* breakbeat class.

These results, when considered with the improved *Amen* breakbeat class performance, indicate the potential of breakbeat classification based on individual drum sounds rather than entire tracks.

When compared with the M-GMM and M-SVM systems, the B-SVM showed improved results for the *Apache* and *Funky Mule* breakbeat classes—77.4% and 93.5%, respectively. Of interest was the accuracy for the *Amen* breakbeat class, which was substantially lower than that for the M-GMM and M-SVM systems. In comparison to the *Apache* and *Funky Mule* breakbeats, the *Amen* is a sonically brighter breakbeat, having a greater idiophone presence above the bass drums than the other two breakbeats. A potential reason for the lower accuracy of the *Amen* breakbeat class in the B-SVM system is the lack of spectral modelling for frequencies not represented by the BFS-based features, which focus only on frequencies approximately between 50–325 Hz. Using both BFS- and MFCC-based features, the BM-SVM system achieved the highest accuracies for the *Amen* breakbeat class (83.9%) and *Apache* breakbeat class (90.3%), the second highest for the *Funky Mule* breakbeat class (87.1%), and the highest overall accuracy for the tested systems (87.1%).

3.2. Evaluation 2: Binary Classification

Whereas the first experiment was designed to test the separability of examples containing three different breakbeats, the second evaluation is performed to establish the viability of breakbeat classification as a binary classification problem, as it is unlikely that any dataset would be limited to examples containing a very small number of breakbeats. Furthermore, development of such a database for training purposes—inclusive of all breakbeats and the variety of manipulations they might take—would also be unlikely. The tested methods are evaluated using two classes: one which contains a particular breakbeat, and the other may contain any breakbeat (or drum from another source) other than the breakbeat in question. As this evaluation is based on a more difficult task than the previous three-class test, we hypothesise that the results will reflect this increase in difficulty, even if the effective baseline will be higher.

To perform our evaluation, we expand the first dataset by focusing on one of the breakbeats used in the first evaluation, and increasing the dataset size such that the number of examples of breakbeat x is comparable to that of not breakbeat x. For this example we chose the *Amen* breakbeat, as it the most well-known breakbeat in the literature on HJDB (e.g., [17]). The dataset contains 148 examples of audio files that exclusively use the *Amen* and 132 examples which do not contain (*non-Amen*). The latter class contains a large number of breakbeats and some examples of HJDB tracks (e.g., early Hardcore tracks) that contain samples from drum machines rather than breaks. All additional breakbeat annotations were made by the first author.

Here we test the two systems presented in Section 2.2. The first is the top-performing configuration of the specialised method from the previous evaluation (BM-SVM), and second, the deep network classification method (BM-DN) which uses the same feature set as the BM-SVM, with one alteration: the PCA stage is not performed to allow for non-linear transformations to occur between layers of the network.

It is important to note that in this second evaluation we have removed the majority voting component. Since we provide results based on each drum assessed (rather than per track), the evaluation is performed using three-fold cross validation in which the dataset is split at the track level to ensure that no drums from a given track appear in both training and testing subsets for a given fold.

The results of the tested methods (BM-SVM and BM-DN) using the extended two-class breakbeat dataset are summarised in Table 2; accuracies are provided as percentages for each breakbeat class, and total accuracies are calculated as the mean across classes per system.

	BM-SVM	BM-DN
Amen	78.4%	**81.1**%
$Non-Amen$	77.1%	**86.1**%
$Avg.$	77.8%	**83.6**%

Table 2: *Accuracies of breakbeat classification systems* (BM-SVM *and* BM-DN) *for binary classification of HJDB examples as either Amen or non-Amen, along with cumulative mean accuracies (Avg.). Bold scores denote the best scores in each breakbeat class and average score.*

As expected, we find a slight drop in performance for the winning system from the previous evaluation, BM-SVM. However, the system performs equally well for both *Amen* (78.4%) and non-*Amen* (77.1%) classes, further demonstrating the appropriateness

of the features selected. The second system, `BM-DN`, outperforms the `BM-SVM` in both the *Amen* (81.1%) and the non-*Amen* (86.1%) classes, demonstrating the benefit of deep learning classification in this context.

4. BREAKBEAT RESEQUENCING

Once a breakbeat has been classified for a given track, this information can then be leveraged for a variety of purposes. One such purpose would be for the musicological task of evaluating the sampling trends of producers in the HJDB genres to learn which breakbeats are more commonly used than others and by which artists, and within which subgenres. Another use of this information is to estimate the arrangement of the breakbeat, which is intended to match the HJDB producer's reordering of the original breakbeat segments. A musician might, for example, decide to create a reordering of the segments from two measures of the breakbeat in the creation of a percussion arrangement in an HJDB track. An overview of this procedure is demonstrated in Figure 2.

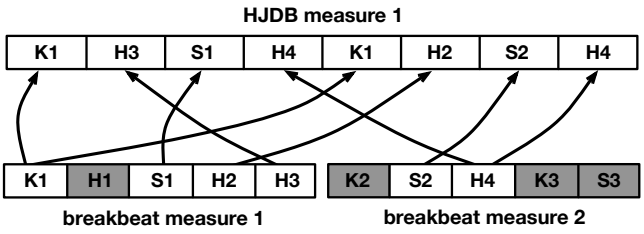

Figure 2: *Example of breakbeat resequencing in practice. The white segments from two separate measures of the breakbeat (bottom) are used to create a single measure of the percussion arrangement in the HJDB track (top). The grey segments are not used.*

To provide an illustrative application of the developed method we can model the percussion arrangement of a target HJDB track using the individual drum segments of the source breakbeat. Unlike the task of drum detection, which requires an explicit declaration of drum class labels, the presented method is able to circumvent this requirement by exploiting the timbral similarity between original breakbeats and the percussion arrangements in tracks that incorporate them. While drum transcription could be used to identify an event as a snare, it is not likely that it could determine that the event was, for example the *second* snare of the *second* measure of the *Amen* breakbeat. We believe that this approach may result in a closer mapping between versions of drums used for a similar purpose (i.e., a well-hit snare drum) due to the knowledge of the timbral character of the percussion. A benefit we see in this approach is that an arbitrary number of drum types could conceivably be included in the pairing, however an obstacle is that the breakbeat being used must be identified. We therefore utilise the breakbeat information extracted in Section 2 to identify the correct audio source (e.g., *Amen* breakbeat), which will serve as the original audio input for the transformation to the target pattern (e.g., PHD & Funky Technicians' *Above and Beyond* (1996)).

The general overview of the process involves the stages of feature extraction, segmentation, and similarity matching to find the best fit segment from the source file that matches that of the target. The feature extraction and segmentation stages are identical for both the source and target files. Features are extracted from each file in a similar method as explained in Section 2.1, with three key differences. First, as we are attempting to match all segments rather than bass drum regions alone, we extract features from entire input audio files. Second, we extend the set of spectral features (i.e., MFCCs and BFS) to include snare drum frequency spectrograms (SFS), which represent frequencies between approximately 430–5000 Hz. In addition to these features, we include the bass-drum detection function Γ_{BD} (Eq. (2)) and a similarly constructed snare drum detection function Γ_{SD} obtained from the SFS.

We then reduce the number of features in the BFS and SFS feature matrices through the application of PCA dimensionality reduction. PCA is applied to reduce the dimensionality of the BFS and SFS matrices to the smallest value that retains 95% variance. The transformation is applied to each feature type (i.e., BFS and SFS) independently to ensure preservation of the drum types exhibited within each feature matrix.

Both source and target signals along associated features are partitioned into individual percussion events by selecting peaks from the summed bass and snare drum detection functions. As mean-segment values will represent each feature dimension, segment boundaries at the beginning and end of each bar are determined through the use of a downbeat detection algorithm [18]. Mean features are extracted from each segment, and features are then normalised across the time axis (i.e., rows) in the source and target representations, separately.

We then evaluate the the similarity of source breakbeat segments to those of the HJDB track through the construction of a cosine similarity matrix M [19]. Selection of a source segment based only on maximum similarity to the target segment resulted in many reconstruction errors. Many of these errors contained large gaps of silence due to an incorrect pairing of short drum hits where a longer more substantial hit was required (e.g., an off-beat *ghost note* in place of a salient snare). We therefore attempted to observe the top r ranking source segments as potential candidates (r = 5). Each segment is weighted by the normalised ratio between the candidate segment length $\rho(r)$ to that of the HJDB segment $\varsigma(n)$ for time segment n as in Eq. (3):

$$w(r) = \frac{\rho(r)/\varsigma(n)}{max(\rho(r)/\varsigma(n))} \tag{3}$$

where w is a series of weights applied to the segments b. The segment exhibiting the maximum interaction after weighting is selected as the winning segment ν for time iteration n:

$$\nu(n) = max(w \cdot b) \tag{4}$$

We replicate the data from the winning source segment $\nu(n)$ in the n^{th} position of the target vector, scaled to match the amplitude of the target segment.

Examples of the presented transformation are made available here.[6] The resequenced breakbeats in the examples sound coherent and similar to the target patterns, albeit without effects such as pitching and distortion, which are generally applied by HJDB producers. We identify three main challenges that will potentially cause errors in the presented method. First and most obvious, errors in breakbeat classification will result in a transformation that will not sound as intended. As the spectral character of drum types differs across breakbeats, incorrect drum type matching may occur. Second, if the onset detection results in spurious or missed notes, then the matching stage will produce additional drum events or

[6] www.music.mcgill.ca/~hockman/projects/breakscience/examples

spaces where drums should occur, respectively. Third, if breakbeats are heavily pitched, one drum type may be matched with another—for example, a bass drum that has been transposed upwards may be matched with snare. Errors due to these factors could be remedied through a semi-automatic method, that would allow users access to the parameters of the transformation. An initial estimation of the percussion arrangement could then be improved through modification of such parameters.

5. CONCLUSIONS

In this paper we present a computational approach for the analysis of breakbeats used in hardcore, jungle and drum & bass recordings. Our chosen approach to this problem is that of music classification, with a specialised procedure of individualised drum classification. To evaluate the plausibility of this solution we first attempted a simplified multi-class problem to determine if our problem formalisation was appropriate. Results of an evaluation with three breakbeat classes demonstrated the effectiveness of specialised processing, which is used to isolate bass drums through suppression of harmonic content and segmentation. We then attempted the more realistic formalisation of a binary classification problem, in which the two classes are tracks that contain a specific breakbeat (*Amen*) and tracks that contain breakbeats other than the specified breakbeat, or in some examples, no breakbeat at all (not-*Amen*). This formalisation is more practical than the multi-class classification approach, as the latter requires ground truth for each breakbeat under analysis. If a HJDB producer wishes to recreate the idiomatic style of a particular breakbeat, then the binary classifier requires ground truth for a single breakbeat. To test the efficacy of this formalisation, we conducted an evaluation with the top-performing model from the multi-class problem (`BM-SVM`) and a deep network classifier `BM-DN`, in which the `BM-DN` outperformed the `BM-SVM`.

For future work, we intend to look into the incorporation of rhythmic features to aid in both the breakbeat classification system, as well as in the breakbeat resequencing analysis. In addition, we will investigate the prospect of deep architecture feature-learning for our classification and transformation.

6. ACKNOWLEDGMENTS

The authors would like to thank the many HJDB producers and DJs who contributed annotations for the datasets used in this research. We would also like to extend our thanks to Alexander Foy, Conor O'Dwyer (Code), Daniel Lajoie (ESB), and Kian Joyner (Godfather Sage/Kian) for providing music from their collections.

7. REFERENCES

[1] S. Reynolds, *Energy Flash: A Journey through Dance Music and Rave Culture*, Soft Skull Press, Berkeley, CA, 2012.

[2] J. Van Balen, "Automatic identification of samples in hip hop music," in *Proceedings of the International Symposium on Computer Music Modelling and Retrieval*, 2012, pp. 544–551.

[3] A. Wang, "An industrial strength audio search algorithm," in *Proceedings of the 4th International Conference on Music Information Retrieval*, 2003, pp. 7–13.

[4] C. Dittmar, K. Hildebrand, D. Gaertner, M. Winges, F. Müller, and P. Aichroth, "Audio forensics meets music information retrieval—a toolbox for inspection of music plagarism," in *Proceedings of the 20th European Signal Processing Conference*, 2012, pp. 1249–1253.

[5] J. L. Whitney, "Automatic recognition of samples in hip-hop music through non-negative matrix factorization," M.S. thesis, University of Miami, 2013.

[6] N. Collins, "Algorithmic composition methods for breakbeat science," in *Proceedings of the International Conference: Music Without Walls? Without Instruments?*, 2001, pp. 21–23.

[7] N. Collins, *Towards autonomous agents for live computer music: Realtime machine listening and interactive music systems*, Ph.D. thesis, University of Cambridge, 2006.

[8] N. Collins, "Influence in early electronic dance music: An audio content analysis investigation," in *Proceedings of the 13th International Society for Music Information Retrieval Conference*, 2012, pp. 1–6.

[9] J. A. Hockman, *An ethnographic and technological study of breakbeats in hardcore, jungle and drum & bass*, Ph.D. thesis, McGill University, 2014.

[10] D. Fitzgerald, "Harmonic/percussive separation using median filtering," in *Proceedings of the 13th International Conference on Digital Audio Effects*, 2010, pp. 203–206.

[11] S. Dixon, "Onset detection revisited," in *Proceedings of the 9th International Conference on Digital Audio Effects*, 2006, pp. 133–137.

[12] M. E. P. Davies, P. Hamel, K. Yoshii, and M. Goto, "Automashupper: Automatic creation of multi-song music mashups," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 12, pp. 1726–1737, 2014.

[13] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 1–27, 2011.

[14] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: A CPU and GPU math expression compiler," in *Proceedings of the Python for Scientific Computing Conference*, 2010.

[15] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. Goodfellow, A. Bergeron, N. Bouchard, D. Warde-Farley, and Y. Bengio, "Theano: New features and speed improvements," in *Proceedings of the NIPS 2012 Deep Learning Workshop*, 2012.

[16] J.-J. Aucouturier and F. Pachet, "The influence of polyphony on the dynamical modeling of musical timbre," *Pattern Recognition Letters*, vol. 28, no. 5, pp. 654–661, 2007.

[17] The Economist, "Seven seconds of fire," *The Economist*, pp. 145–146, December 2011.

[18] J. A. Hockman, M. E. P. Davies, and I. Fujinaga, "One in the jungle: Downbeat detection in hardcore, jungle, and drum & bass," in *Proceedings of the 13th International Society of Music Information Retrieval Conference*, 2012, pp. 169–174.

[19] M. Cooper and J. Foote, "Automatic music summarization via similarity analysis," in *Proceedings of the 3rd International Symposium on Music Information Retrieval*, 2002, pp. 81–85.