Adaptive Mapping of Sound Collections for Data-driven Musical Interfaces

Gerard Roma
CeReNeM
University of Huddersfield
g.roma@hud.ac.uk

Owen Green
CeReNeM
University of Huddersfield
o.green@hud.ac.uk

P. A. Tremblay
CeReNeM
University of Huddersfield
p.a.tremblay@hud.ac.uk

ABSTRACT

Descriptor spaces have become an ubiquitous interaction paradigm for music based on collections of audio samples. However, most systems rely on a small predefined set of descriptors, which the user is often required to understand and choose from. There is no guarantee that the chosen descriptors are relevant for a given collection. In addition, this method does not scale to longer samples that require higher-dimensional descriptions, which biases systems towards the use of short samples. In this paper we propose novel framework for automatic creation of interactive sound spaces from sound collections using feature learning and dimensionality reduction. The framework is implemented as a software library using the SuperCollider language. We compare several algorithms and describe some example interfaces for interacting with the resulting spaces. Our experiments signal the potential of unsupervised algorithms for creating data-driven musical interfaces.

Author Keywords

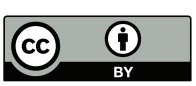
Dimensionality reduction, feature learning, information visualization

CCS Concepts

•Computing methodologies → Dimensionality reduction and manifold learning; •Applied computing → Sound and music computing;

1. INTRODUCTION

Interacting with collections of digital audio samples is nowadays part of many music creation workflows. Samples can come from a diversity of sources: from loops crafted for dance music, to field recordings, personal improvisation, commercial music releases, or instrument samples. Often, a collection of samples will constitute the creative material for one or several compositions or performances, imprinting their specific sound. One such collection can be defined as a *corpus*. Within this paper we refer to a sound corpus as a collection of samples that could be united by as little as a practitioner putting them together with some musical intent. In practice, sound corpora are often made of samples that are similar in some way, or share a common origin.



NIME'19, June 3-6, 2019, Federal University of Rio Grande do Sul, Porto Alegre, Brazil.

While there has been significant research on signal processing and machine learning for dealing with audio, particularly in fields such as music information retrieval, the repertoire of tools currently available to music makers for dealing with sound corpora is still limited, requiring manual annotation, bookkeeping and editing.

A significant amount of research has followed the adaptation of corpus-based concatenative synthesis and musical mosaicing to open-ended music creation systems [16, 1]. The interface is typically based in an interactive visualization of the corpus. These systems suffer from several limitations, partly inherited from the roots in realistic synthesis.

First, they often rely on specific descriptors (e.g. pitch, spectral centroid), typically requiring the user to choose two or three of them from a pre-defined set. This requires an understanding of concepts related with signal processing and psychoacoustics. Moreover, there is no assurance that a given sound corpus will have an interesting variation along a given set of descriptors. For example, a pitch descriptor may be irrelevant for a corpus obtained from environmental sounds. In general, a corpus may have its own sonic dimensions beyond a particular offer of descriptors.

Second, such descriptors are typically obtained from a frame-level representation, which means they may vary significantly over time. A single value (typically the average over a sequence of frames) may be relevant for very short sound, but it will not be as useful for longer samples. Describing a longer sound (e.g. in the order of a few seconds) typically involves computing several statistics or other ways of describing a trajectory, which increases the number of parameters needed to represent each sound.

As an attempt towards solving both these issues, in this paper we propose to automate the analysis so that the generation of the interaction space is driven by the corpus rather than by pre-existing assumptions about pitch or timbre. The proposed framework allows learning both the base short-term features, and the mapping of high-dimensional summaries of sounds to a low-dimensional space that can be used in interactive applications. The framework is implemented as a library in the SuperCollider language.

The rest of the paper is organized as follows. In the next section we briefly review existing work related with the use of feature spaces for interactive applications. We then review dimensionality reduction algorithms that have been previously applied to sounds. In Section 3, we describe the proposed method for automatically generating feature spaces from sound corpora. In Section 4, we compare several dimensionality reduction techniques and two different base features in a visualization experiment and a subjective reflective practice experiment. In Section 5, we describe several interfaces for interacting with the generated spaces, and in Section 6 we draw some conclusions.

2. BACKGROUND

2.1 Musical interaction with sound spaces

The application of *sound spaces* (which can be defined as spatial representations of sound corpora through content-based descriptors) to playable musical interfaces was presented in [15], using two-dimensional plots and indirect 2D control surfaces. Our work follows a similar direction, but we explore a bottom-up approach where the space is automatically generated. We also demonstrate several interfaces based on currently popular devices with multi-touch screens. Our use case should be distinguished from more functional sound browsing interfaces (e.g. [8]), in that we aim to create interfaces that can be directly used in musical composition and performance. Also, our framework includes tools for adaptive segmentation of the corpus, supporting the whole path from a bunch of recordings to an interactive sound space.

In the next section we overview the most common dimensionality reduction algorithms that have been used for sounds. Some of these were compared using algorithmic measures in [2]. Our work is more concerned with assessing the musical affordances of these different ways for visualizing corpora.

2.2 Dimensionality reduction

A crucial element of our system that has seen less research in the context of creative interactive systems is a dimensionality reduction algorithm that projects a high-dimensional vector, resulting from summarizing a sequence of frame-level descriptors, to a lower-dimensional space that can be used to interact with the corpus. Let X be the matrix obtained by stacking the set of points $x_i ... x_n$ representing the sounds in the corpus. The points typically do not fully span the space. We seek to obtain a low-dimensional set of vectors $y_i ... y_n$ that preserves the original distances between points. Many algorithms have been used, mainly from machine learning and data visualization communities, for visualizing collections of sounds.

2.2.1 Principal component analysis (PCA)

PCA [9] is perhaps the most widely used data reduction technique. It maps data to a set of uncorrelated features that are ranked in terms how much of the variance in the input data each one accounts for. These features can be obtained by finding the eigenvectors corresponding to the largest eigenvalues of the covariance matrix of $XX^\top$ (where $.^\top$ is the matrix transpose).

2.2.2 Multidimensional scaling (MDS)

MDS [19] works by mapping a similarity matrix of pairwise distances in the original space Dx to a matrix of distances in the low-dimension space Dy. The mapping can be expressed as an optimization problem that minimizes a *stress* function representing the difference between Dx and Dy,

$$Stress = \sum_{i \neq j} (Dy_{ij} - Dx_{i,j})^2 \tag{1}$$

MDS techniques are widely used in statistics, and have been used notably in perceptual studies of timbre [10].

2.2.3 Isomap

Isomap [18] can be regarded as an extension of MDS, but instead of trying to directly reproduce distances from the input similarity matrix, it derives new distances by creating a k-nearest neighbors graph (k-NNG) of the input points. New distances are then found from the shortest paths in the graph, and finally MDS is used to try and reproduce these distances in the lower-dimensional space. Isomap was used in [3] to derive mappings between synthesis parameters and audio features.

2.2.4 Force-directed graph layouts

Algorithms used generally to visualize graphs can be used to represent k-NNGs of audio samples. A common strategy is to use a simulation of attraction and repulsion forces based on the edges connected to each node. One of the most popular is the Fruchterman-Reingold (FR) algorithm [6]. Graph layout algorithms were used for exploring sounds in the Freesound database [13], as well as for navigation inspired by corpus-based synthesis [17].

2.2.5 t-SNE

t-SNE [21] is a more recent, popular algorithm which is specialized for visualizing high-dimensional data. It uses a probabilistic model of similarity, which expresses the likelihood of a point, x_i, neighboring a second point, x_j, as:

$$p_{ij} = \frac{\exp(-||x_i - x_j||^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-||x_i - x_k||^2 / 2\sigma_i^2)}, \tag{2}$$

where σ_i is the standard deviation of the Gaussian centered at x_i. A 'perplexity' parameter allows the user to control the values of σ_i. Similarity between points $y_i ... y_n$ in the lower-dimensional space is expressed as a probability using a Student t-distribution as:

$$q_{ij} = \frac{(1 + ||y_i - y_j||^2)^{-1}}{\sum_{k \neq i} (1 + ||y_k - y_i||^2)^{-1}}. \tag{3}$$

t-SNE then tries to minimize the difference between the joint probability distribution of the input points, P, and the joint probability distribution of the mapped points, Q by minimizing the Kullback-Leibler divergence:

$$D_{KL}(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{4}$$

using gradient descent. This algorithm was used also for Freesound sounds in [4], and for visualizing textural sounds in [7].

2.2.6 Self-organizing Maps (SOM)

A SOM [22] uses an artificial neural network to learn a mapping between X and a lower, usually 2-dimensional arrangement of points. The network is trained with a 'winner-takes-all' approach so that training examples become associated with one point in the output map. Winning neurons during training are selected on the basis of the Euclidean distance between the input training vector, x, and a neuron's weight vector, w. The weights of each neuron are then updated as a function of the distance of that neuron from the training input:

$$w_n = w_n + \eta(t) h(i) (x - w_n) \tag{5}$$

where $\eta(t)$ is the learning rate, and $h(i)$ is a function that returns high values for neurons close to the winning neuron in the new space, and lower values for more distant ones. Since the SOM collapses input points to one of the points in the output grid, a common strategy is to add some jitter so that outputs cluster around the grid point corresponding to their winning neuron. SOMs were used to visualize drum sample libraries in [11] and sound effect collections in [8].

3. GENERATING ADAPTIVE SPACES

In this section we describe the proposed framework for generating musical interfaces from an audio corpus. The central piece of the implementation is a SuperCollider library that

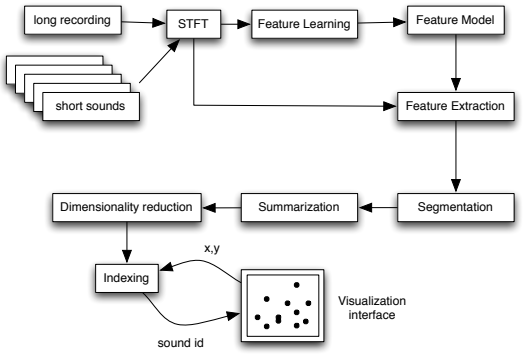


Figure 1: Adaptive sound space mapping workflow

supports creative workflows for analysis, segmentation and mapping of sound coropra to low-dimensional spaces. The resulting databases can be used for music creation and/or performance by visual interaction either from SuperCollider or from a tablet computer by rapid development of web interfaces. The library allows the use of several plug-in objects implemented in the Fluid Decomposition Toolbox [20], as well as interfacing with Python for using further analysis and dimensionality reduction algorithms. We now describe the main elements of the full workflow.

3.1 Overview

The process is summarized in Figure 1. The input of the system can be either a long recording or a collection of samples, or any combination. All the audio in the corpus is analyzed through a short-time Fourier transform (STFT), and the resulting frames used for learning a frame-level feature. The resulting model then produces a feature vector for each spectral frame in each input file. A segmentation algorithm may then be used to extract units from longer recordings. For each segment, frame-level features are then aggregated to a higher-dimension segment-level descriptor. A dimensionality reduction algorithm is then applied to project the data to a manageable number of dimensions. The resulting vectors are indexed using a KD-tree for rapid searching from client applications.

3.2 Feature extraction

The main challenge for the feature extraction step is the potential variety of input signals. In order to facilitate different representations, our system allows the use of several features. By default, a neural autoencoder is used to compress the STFT frames into lower dimension vectors. The model is trained for each corpus, so that the user can choose the number of input and output dimensions by setting the STFT parameters, and the number of hidden units. In our experiments (Section 4) we used two hidden layers resulting in a hidden representation of 13 dimensions, in order to compare with the conventional size of Mel-Frequency Cepstrum Coefficients (MFCCs). The architecture is based on contractive autoencoders [12].

3.3 Segmentation

Common strategies for splitting longer recordings into playable corpora are fixed-length segmentation and onset detection. In order to obtain a more versatile segmentation, we used Foote's classic novelty algorithm [5] which can be used with any frame-level feature. This allows us to adapt to the feature used in the previous step. For a given sequence of feature vectors, the algorithm computes a cosine similarity matrix and slides a 2D checkerboard kernel through the diagonal, which produces a novelty curve. In order to obtain the segments, the algorithm requires choosing the size of the kernel and a threshold. We found these parameters to be musically useful for determining a general length of the resulting segments, while still adapting to the input material. For analyzing material with traditional music structures, onset detection can also be used.

3.4 Summarization

In order to compare different segments in the corpus, the time series of frame-level features need to be summarized into a common space. Our system currently uses basic statistics (mean, standard deviation, minimum and maximum), which are applied to both the raw features and their first derivative. For thirteen dimensions in the frame-level feature, this renders a vector of 104 dimensions.

3.5 Dimensionality reduction

The final vector for each unit in the corpus is obtained by projecting it using one of the algorithms outlined in Section 2.2.

4. COMPARING ALGORITHMS

Many dimensionality reduction algorithms have been used in the literature related to sounds, but there is in general little guidance as to which of them would work in a particular situation related to corpus-based music creation. While some empirical quality metrics have been proposed (see e.g. [2]), there is no evidence that they are particularly suited for creative tasks. Moreover, they are often related in some way to the measures the algorithms try to optimize (i.e. how close the low-dimensional space reproduces similarities in the original space), which may be misleading with respect to the qualities of different algorithms.

In this paper, we are interested in the musical affordances offered by both the dimensionality reduction algorithms and the feature extraction algorithms. In this context, it is difficult to imagine one particular algorithm being "the best". Instead, the choice will generally be rather subjective, and dependent on the type of sound materials and musical objectives. Hence, our assessment is based on two qualitative experiments: a visualization experiment and a subjective listening experiment using the proposed system. Through these, we identified three main dimensions of interest. First, we found the main musical affordance to be the separation of sounds into clusters of perceptually similar sounds. Second, a practical interest is the efficiency in utilization of screen space. Finally, computational efficiency has also a significant impact in the usability of the system in creative workflows, although this can be influenced by implementation aspects outside of the scope of this work.

4.1 Visualization

In addition to similarities in the original space, an important requirement for dimensionality reduction is that it helps uncover existing structures in the data. In this sense, dimensionality reduction is often used as pre-processing for classification tasks, and classification of labelled data can be used to assess the quality of the low-dimensional representations. We devised the visualization experiment using three annotated datasets in order to see how the different algorithms would allow distinguishing different classes artificially mixed in a dataset, in addition to the different shapes generated by each algorithm. The first dataset was composed of 750 drum samples obtained from commercial sampler programs. The second dataset was obtained as a subset of the SOL database of musical instrument articulation sounds that is distributed with the Orchids framework

for computer-aided orchestration[1]. We selected five violin articulations that have more than 100 examples, which resulted in 1239 sounds. Finally, we created a dataset of environmental sounds from the Urban Sounds dataset [14]. Since these sounds often have background noise and are only weakly related to the label, we selected 4 classes that seemed easier to distinguish.

For each dataset, we analyzed all the sounds using both MFCC and autoencoder (AE) features, and summarized each file as described in Section 3.4. In both cases, we used windows of 20 ms and hops of 10 ms for the STFT. The resulting vectors of 104 dimensions were then reduced to 2 by each of the algorithms. We used the implementations available in the *scikit-learn* Python library[2], along with the *MiniSom*[3] library for the SOM algorithm, and *Networkx*[4] for the FR algorithm. We manually tweaked the parameters of each algorithm to the setting with best results.

The results can be seen in Figures 2 and 3 (the plots had to be reduced due to space constraints, for larger images, we invite the reader to consult the companion page of this paper[5]).

From visual inspection, the separation of classes is generally worse with PCA and MDS. MFCC features generally result in better separation of classes but also more irregular shapes, while AE features result in similar local distributions with respect to the classes, but more efficient spatial distribution. For both features, t-SNE, SOM and FR (labelled as *Graph*) tend to have a better balance between separation and space utilization.

4.2 Assessing Musical Affordances

In order to develop a more concrete understanding of the difference between features and algorithms in a practical situation, we experimented creating several playable sound spaces, this time with several unlabelled sound collections, using the SuperCollider implementation. We were interested in how the parameters of each algorithm would affect the visualization in an interactive setting. In general, we observed that, compared with CataRT [16], the resulting visualizations produce more meaningful clusters, and perceptually similar sounds were placed together. On the other hand, the axes did not make as much sense globally as in CataRT but, unless the corpus had a very interesting variation along a given descriptor, it was preferable to have areas of similar sounds. When dealing with collections of pitched instrument sounds, we noted that sounds with overlapping harmonics (i.e. same pitch, fifth or octave) tended to cluster together. This still happened with MFCC features, but it was more salient with AE features. While t-SNE generally gave good results, the execution time was significantly slower. Meanwhile, the Isomap and FR algorithms ran faster and had a common behavior with respect to the number of neighbors. Very small numbers (below 5) were 'risky', resulting in perceptually close clusters but low space utilization, and variable distance between similar clusters. Between 5 and 10, both algorithms produced paths that would have meaningful perceptual changes along their main direction, but with still low space occupation. Increasing the number of neighbors to between 10 and 50 was an overall safer choice for space efficiency, while higher numbers started to produce less meaningful visualizations. These values can be related to the size of the dataset, which in our case oscillated around a thousand sounds. With respect to the SOM, its main interest lies in the ability to cluster sounds in a 2D grid. In this sense, the space utilization could be controlled by the amount of jitter. However, the main parameter of interest was the number of output units. The difficulty of this parameter is that it 'saturates', and at some point the algorithm will stop populating output units. At the same time, high values ensure all sounds in the same cluster are related.

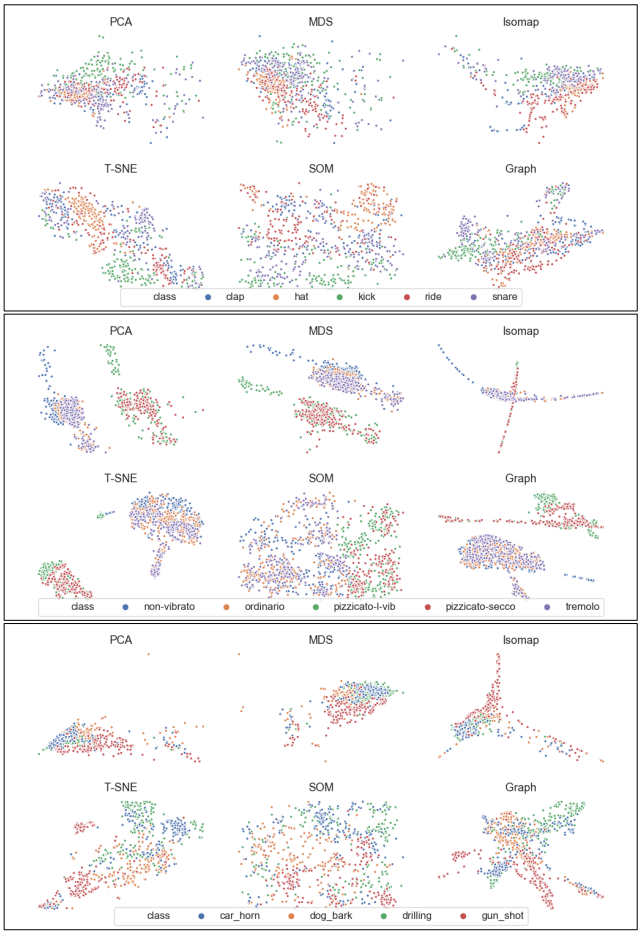


Figure 2: Visualization of each dataset (top: drums, middle: SOL, bottom: urban) using MFCC features

5. MUSICAL INTERFACES

In the previous sections we have described a framework for adaptive mapping of sound corpora to low-dimensional spaces. Our goal is to facilitate the creation of musical interfaces driven by the selection of sounds, with minimal assumptions. Given the current ubiquity of touchscreen computers, which are increasingly being used as interfaces for music creation applications, we were interested in the possibility of devising touchscreen-based musical instruments driven by sound corpora. We extended our framework using the supercollider.js[6] client library. This allowed us to explore rapid development of web-based interfaces that can be quickly deployed to current tablet computers. Analysis, segmentation and mapping can be done interactively within SuperCollider. The resulting dataset is then exported to the web application. The web applications run on a tablet computer and communicate with a SuperCollider server that holds all the sounds in memory. This provides a scalable and

[1] http://forumnet.ircam.fr/product/orchids-en/
[2] https://scikit-learn.org/
[3] https://github.com/JustGlowing/minisom
[4] https://networkx.github.io/
[5] http://flucoma.org/NIME-2019/
[6] https://github.com/crucialfelix/supercolliderjs

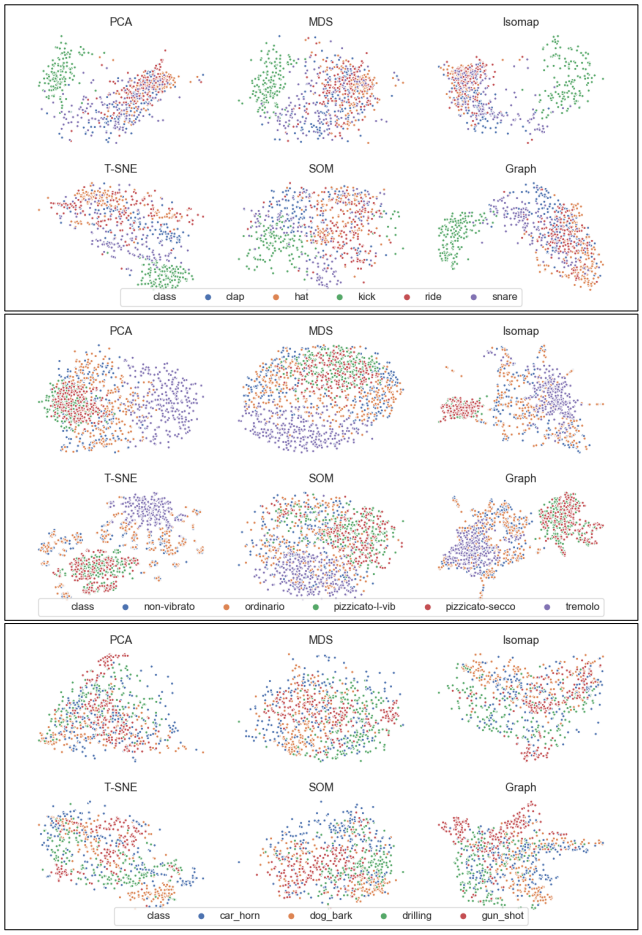


Figure 3: Visualization of each dataset (top: drums, middle: SOL, bottom: urban) using autoencoder features

convenient setup for devising musical instruments with large corpora. The code for this system can be obtained from `https://github.com/flucoma/FluidCorpusMap`. We now describe three example instruments that show its potential for creating data-driven musical interfaces.

5.1 Multi-touch scatterplot

The first example is based on the tradition of using a scatterplot to represent the corpus [16][1]. Here, we used t-SNE to map the autoencoder features to three dimensions. The third dimension was mapped to color. The size of the circles represents the length of the file. An example can be seen in Figure 4, using a corpus of 700 sounds of bowed cardboard box. Using a multi-touch screen allows up to 10 simultaneous voices, turning any collection of samples into a versatile sampler instrument.

5.2 Data-driven Multi-slider

In order to explore the possibilities of using more than two or three dimensions, we developed a multi-slider interface, where each slider represents an axis of a multi-dimensional space. The histogram of the points in each dimension is visualized in each slider (Figure 5). The mapping was again obtained with t-SNE. The result is that the relations between axes are disconnected, as the space is allocated independently for each slider. We experimented with a dataset of 500 sounds corresponding to a single note from many different instruments in the SOL collection, using a granular synthesizer reading from the buffer selected by the multi-

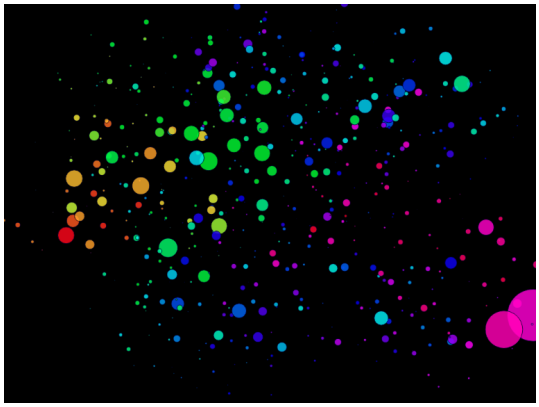


Figure 4: Multi-touch scatterplot

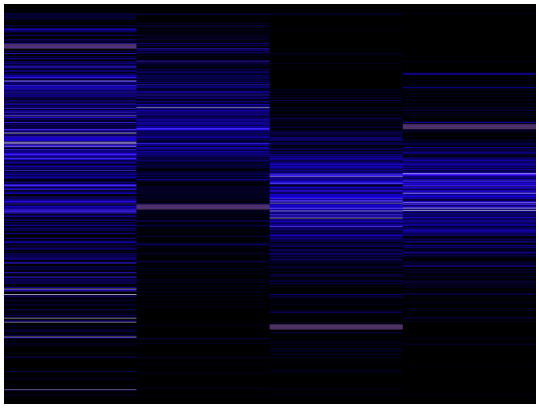


Figure 5: Data-driven multislider

slider at any time. While it was not very easy to make sense of each dimension, the sliders worked as a search interface. In this sense, it was easy to learn the locations of particular sonorities. Also, the histogram visualization made it easy to control the amount of change produced by a given gesture. This interface was also less visually driven than the scatterplot, allowing the user to pay more attention to the resulting sounds.

5.3 Self-organized drum machine

Finally, given the current popularity of grid-based interfaces, we were interested to try the mapping of two-dimensional grids generated by SOMs. In order to visualize the sounds of each cluster, we used a second dimensionality-reduction step using the FR algorithm. The interface works like a regular drum machine (Figure 6), pads can be locked for programming a pattern, or unlocked to become 2D sliders for varying their sound. The result is consistent with common rhythm programming experiences, except that the two-dimensional map provides a more intuitive interface than just scrolling through a list, allowing the user to easily navigate a large collection of sounds while playing a pattern in real time.

6. CONCLUSIONS AND FUTURE WORK

In this paper we have presented a novel framework for adaptive mapping of sound collections that facilitates the development of data-driven musical interfaces. The framework is implemented as a software library, including all the necessary elements for obtaining playable interfaces from a set of audio files. We have described three example interfaces that demonstrate the potential of our approach. Given the number of options for the dimensionality reduction step, which

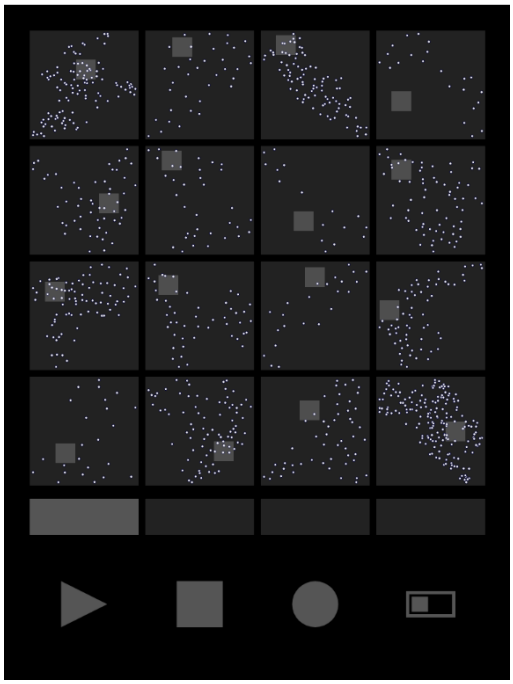


Figure 6: Self-organized drum machine

has not been thoroughly investigated for the case of creative interactive systems, we have mainly focused on qualitative evaluation of different dimensionality reduction algorithms.

As future work, we hope to review the constraints of this approach with respect to the number and duration of sounds in the collection. Given the simplicity of the feature learning step, we are also considering more advanced pipelines.

7. ACKNOWLEDGMENTS

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 725899)

8. REFERENCES

[1] G. Coleman. Mused: navigating the personal sample library. In *Proceedings of the International Conference on Computer Music (ICMC)*, 2007.

[2] S. Dupont, T. Ravet, C. Picard-Limpens, and C. Frisson. Nonlinear dimensionality reduction approaches applied to music and textural sounds. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, 2013.

[3] S. Fasciani and L. L. Wyse. Adapting General Purpose Interfaces to synthesis Engines using Unsupervised Dimensionality Reduction Techniques and inverse Mapping from Features to parameters. In *Proceedings of the International Conference on Computer Music (ICMC)*, 2012.

[4] F. Font and G. Bandiera. Freesound Explorer: Make Music While Discovering Freesound! In *Proceedings of the Web Audio Conference (WAC)*, 2017.

[5] J. Foote. Automatic audio segmentation using a measure of audio novelty. In *Proceedings of the IEEE International Conference on Multimedia and Expo.*, 2000.

[6] T. M. J. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. *Software: Practice and Experience*, 21(11):1129–1164, 1991-11.

[7] T. Grill and A. Flexer. Visualization of perceptual qualities in textural sounds. In *Proceedings of the International Conference on Computer Music (ICMC)*, 2012.

[8] S. Heise, M. Hlatky, and J. Loviscach. SoundTorch: Quick Browsing in Large Audio Collections. In *Proceedings of the 125th Convention of the Audio Engineering Society*, 2008.

[9] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417–441, 1933.

[10] S. McAdams. Perspectives on the Contribution of Timbre to Musical Structure. *Computer Music Journal*, 23(3,):85–102, 1999.

[11] E. Pampalk, P. Hlavac, and P. Herrera. Hierarchical organization and visualization of drum sample libraries. In *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, 2004.

[12] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pages 833–840. Omnipress, 2011.

[13] G. Roma and X. Serra. Music performance by discovering community loops. In *Proceedings of the Web Audio Conference (WAC)*, 2015.

[14] J. Salamon, C. Jacoby, and J. P. Bello. A dataset and taxonomy for urban sound research. In *Proceedings of the ACM International Conference on Multimedia*, 2014.

[15] D. Schwarz. The Sound Space as Musical Instrument: Playing Corpus-Based Concatenative Synthesis. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, 2012.

[16] D. Schwarz, G. Beller, B. Verbrugghe, and S. Britton. Real-Time Corpus-Based Concatenative Synthesis with CataRT. In *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, 2006.

[17] D. Schwarz, N. Schnell, and S. Gulluni. Scalability in Content-Based Navigation of Sound Databases. In *Proceedings of the International Computer Music Conference (ICMC)*, 2009.

[18] J. B. Tenenbaum. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science*, 290(5500):2319–2323, 2000-12-22.

[19] W. S. Torgerson. Multidimensional scaling: I. Theory and method. *Psychometrika*, 17(4):401–419, 1952-12-01.

[20] P. A. Tremblay, O. Green, G. Roma, and A. Harker. From collections to corpora: Exploring sounds through fluid decomposition. In *Proceedings of the International Conference on Computer Music (ICMC)*, 2019 (in press).

[21] L. van der Maaten and G. Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[22] J. Vesanto. SOM-Based Data Visualization Methods. *Intelligent Data Analysis*, 3:111–126, 1999.