Computer-assisted corpus exploration with UMAP and agglomerative clustering

James Bradbury[1]

University of Huddersfield
james.bradbury@hud.ac.uk

Abstract. In this article, I outline the development of corpus navigation software for composing with a large corpus of samples generated by 'moshing' files. The paper will discuss the technical implementation and rationale for decisions made in the development process as well as touching on some compositional applications.

Keywords: dimension reduction, corpus navigation, clustering

Introduction This article explicates the development process of software for computer-assisted corpus exploration. While there is existing software grounded in similar interests such as *CataRT* (Schwarz, Beller, Verbrugghe, & Britton, 2006), *FluidCorpusMap* (Roma, Green, & Tremblay, 2019) or AudioStellar,[1] none of these tools integrate with my artistic practice, which employs mixture of computer-aided techniques to compose with collections of textural sounds. This software was specifically developed for composing an album of works, to which a pre-release can be found at `https://jamesbradbury.xyz/docs/reconstruction_error`. The software is available for modification and use, so this work could extend into other contexts where machine learning thrives, such as in fully autonomous compositional programs, recommendation tools or programs for intelligently recombining musical material. The code is available at `https://www.github.com/jamesb93/data_bending`.

Creating a synthetic corpus with "moshing" The corpus is generated by "moshing", where raw data is converted into audio files. Moshing was orchestrated with a bespoke command line tool.[2] Overall, it produces a diverse corpus of sounds ranging from purely noisy to morphologically interesting, almost humanly composed gestures.

Segmentation Once the corpus is generated, each corpus item is segmented. My segmentation rationale was guided by listening and I settled on the model that demarcations aligned with significant spectral shifts. After experimenting with segmentation algorithms from the Librosa (McFee et al., 2015) library, such as Laplacian segmentation[3] and observing spectrograms of corpus items, I sought

[1] `http://audiostellar.xyz`
[2] `https://www.github.com/jamesb93/mosh`
[3] `https://librosa.org/doc/latest/auto_examples/plot_segmentation.html`

an algorithm that could be tuned to detect these spectral boundaries. Using Jonathon Foote's novelty algorithm (Foote, 2000), I produced segmentations on several test items that aligned well with perceptually notable spectral change. I applied the settings from these tests in a pass over every corpus item, as I did not want to spend an indefinite amount of time solving a segmentation problem, accepting that some files might be poorly segmented.

Analysis The next step was to produce analysis that could be supplied to further processes. I planned to use features such as spectral centroid, loudness, pitch and spectral flatness- however, there are issues when working with such features. First, they are often calculated as statistical summaries of frame-by-frame analysis requiring careful sanitisation. Ben Hackbarth has shown promising results with *AudioGuide* (Hackbarth, Schnell, & Schwarz, 2010) using perceptual weighting of frames to solve such problems. Furthermore, choosing an appropriate statistical summary can favour certain perceptual features and influence the meaning of the descriptors. Second, scaling of descriptors can be problematic and reconciling various scales can change how a corpus is represented. This led me to use MFCCs for analysis as they are robust against differences in loudness and are capable of differentiating the textural characteristics of sounds. For each sample, an MFCC analysis was conducted using FluCoMa's implementation with a window size of 2048, hop size of 1024, 40 melbands and 13 coefficients. Seven statistics- mean, standard deviation, skewness, kurtosis, minimum, median and the maximum, are taken for each band along with two derivatives. This was an effort to capture the morphology of the sound. These statistical summaries are flattened to one-dimension and each column of these vectors is standardised.

Dimension Reduction While MFCC values are strictly defined, they are hard to interpret and to relate to higher level musical characteristics. In similar work, various dimension reduction techniques have been used to produce compressed representations of data. Examples are Stefano Fasciano (Fasciani, 2015), *FluidCorpusMap* (Roma et al., 2019), *Flow Synthesizer* (Esling, Masuda, Bardet, Despres, & Chemla-Romeu-Santos, 2019) and Thomas Grill (Grill & Flexer, 2012). Using the algorithm "Uniform Manifold Approximation and Projection" (UMAP) (McInnes, Healy, & Melville, 2018), the MFCC dimensions was reduced from 273 to 2 to support visualisation. A strength of UMAP is its potency for capturing non-linear features compared to algorithms such as Principal Component Analysis. Furthermore, UMAP can be coerced to favour global or local structure through the "minimum distance" (*mindist*) and "number of neighbours" (*n_neighbours*) parameters, useful for manipulating the projection to favour various spatial relationships.

Clustering The reduced data was then clustered to understand how corpus items were projected onto the manifold. Using a hierarchical clustering algo-

rithm,[4] I ran a clustering pass with 250, 500 and 1600 clusters. These were chosen from intuition and the relationship of these amounts to the total items.

Application of Outputs Visualising the characteristics of the UMAP projections, the topology becomes a source of inspiration from the location, shape and relationships between clusters of samples. In particular, sample clusters located away from the main body of the projection became sites of interest that were investigated further through manual audition. This was the first method of exploration that demonstrated the success of the combined analysis in perceptually mapping out the corpus items.[5]

The clustering outputs were used in composition in a direct manner by juxtaposing clusters for musical effect. This is evidenced in the first track (340685107feisraebbaatsaedisn.sqlite) where clusters of impulse-based material are superimposed to create phasing and micro-rhythmic patterns. These structures are formed by the sample's memberships to those clusters and otherwise would have to be manually arranged or organised through other means. Clustering somewhat ensures that each layer in the texture is perceptually homogenous and allows me to work with the notion of a "cluster" as unified compositional material.

Prior to clustering, manual auditioning was used to categorise some types of sounds generated from moshing. By knowing the characteristics of samples that were manually categorised, one can speculate that samples belonging to the same computationally calculated cluster should be texturally similar. This became the basis of the final track of the album which features "outlier" sounds from the corpus that have chaotic and noisy spectra.

Conclusion Much existing corpus software is oriented around source/target models and does not support workflows involving digital audio workstations. This article presents techniques for analysing large sound collections and composing mostly cluster data. While working like this suits me, there are advantages that could benefit other practices based on computer-assisted workflows especially for those who use sample-based materials. Most importantly, this software brings the process of corpus exploration closer to the compositional process. Clustering aligns closely with my preferences for dealing with compositional material as composite groups, with the computer seemingly able to suggest organisations through data-driven processes. In the future, this system will be further developed under the banner "FTIS" (Finding Things in Stuff). Work has already begun on developing a command-line tool for which similar processes as found in this paper can be coordinated easily and with speed. I intend to include more computer-assisted processes for composing with these types of analyses, enabling additional methods of exploring sound collections with the aid of content-aware processes. This can be found at `https://www.github.com/jamesb93/ftis`.

[4] `https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html`

[5] Interactive UMAP projections are viewable at `https://umapaudio.onrender.com/src/interactive_scatter.html`.

References

Esling, P., Masuda, N., Bardet, A., Despres, R., & Chemla-Romeu-Santos, A. (2019, July). Universal audio synthesizer control with normalizing flows. *arXiv:1907.00971 [cs, eess, stat]*. Retrieved 2020-08-02, from `http://arxiv.org/abs/1907.00971` (arXiv: 1907.00971)

Fasciani, S. (2015). Interactive computation of timbre spaces for sound synthesis control. *Sound and Interactivity*, *20*, 69.

Foote, J. (2000). Automatic audio segmentation using a measure of audio novelty. In *2000 ieee international conference on multimedia and expo. icme2000. proceedings. latest advances in the fast changing world of multimedia (cat. no. 00th8532)* (Vol. 1, pp. 452–455).

Grill, T., & Flexer, A. (2012). Visualization of perceptual qualities in textural sounds. In *The international computer music conference* (pp. 589–596).

Hackbarth, B., Schnell, N., & Schwarz, D. (2010). *Audioguide: a framework for creative exploration of concatenative sound synthesis* (Tech. Rep.). Citeseer.

McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., & Nieto, O. (2015). Librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference* (Vol. 8, pp. 18–25).

McInnes, L., Healy, J., & Melville, J. (2018, December). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *arXiv:1802.03426 [cs, stat]*. Retrieved 2020-08-01, from `http://arxiv.org/abs/1802.03426` (arXiv: 1802.03426)

Roma, G., Green, O., & Tremblay, P. A. (2019). Adaptive mapping of sound collections for data-driven musical interfaces. , 313–318.

Schwarz, D., Beller, G., Verbrugghe, B., & Britton, S. (2006). Real-time corpus-based concatenative synthesis with catart. In *The 9th international conference on digital audio effects* (p. 279).