Orangeburg, NY, USA
Neuromodulation, Nathan Kline Institute for Psychiatric Research, Computational Neuroimaging Lab, Center for Biomedical Imaging and Craddock1,2
College, New York, NY, USA;4Max Planck Research Group for Processing:

The 200 fMRI time courses were analyzed to extract pitch, CC200 atlas [4] to downsample voxels to 200 regions-of-interest. Different strategies. To reduce the data dimensionality, we used the 10 individual datasets preprocessed using C-PAC pipeline [3] with 4 by the Preprocessed Connectomes Project [2]. We randomly chose Data
til library [https://code.google.com/p/midiutil/].

musical form. Our project is implemented in Python using the midiu-
input, we use basic rules of music theory to transform the data into dictable hills and valleys. People with musical training may notice

Introduction

2016,
GigaScience

Table 3 (abstract A8). Nine pipelines to be implemented

| EPI | T1 | T2 | Diffusion Field Map | Reverse Phase Encode EPI |
|-----|----|----|---------------------|--------------------------|
| N   | N  | N  | 0                   | 0                        |
| N   | N  | 0  | 1                   | 0                        |
| N   | 0  | N  | 1                   | 0                        |
| N   | N  | N  | 0                   | N                        |
| N   | N  | 0  | 0                   | N                        |
| N   | 0  | N  | 0                   | N                        |
| N   | N  | 0  | 0                   | 0                        |
| N   | N  | 0  | 0                   | 0                        |
| N   | 0  | N  | 0                   | 0                        |

A9

Generating music with resting-state fMRI data
Caroline Froehlich1, Gill Dekel3, Daniel S. Margulies4, R. Cameron Craddock1,2
1Computational Neuroimaging Lab, Center for Biomedical Imaging and Neuromodulation, Nathan Kline Institute for Psychiatric Research, Orangeburg, NY, USA;2Center for the Developing Brain, Child Mind Institute, New York, NY, USA;3City University of New York-Hunter College, New York, NY, USA;4Max Planck Research Group for Neuroanatomy & Connectivity, Max Planck Institute for Human Cognitive and Brain Sciences, Leipzig, Germany

Correspondence: Caroline Froehlich (cfrohlich@nki.rfmh.org) – Computational Neuroimaging Lab, Center for Biomedical Imaging and Neuromodulation, Nathan Kline Institute for Psychiatric Research, Orangeburg, NY, USA

GigaScience 2016, 5(Suppl 1):A9

Approach

Data We used open rsfMRI from the ABIDE dataset [1] preprocessed by the Preprocessed Connectomes Project [2]. We randomly chose 10 individual datasets preprocessed using C-PAC pipeline [3] with 4 different strategies. To reduce the data dimensionality, we used the CC200 atlas [4] to downsample voxels to 200 regions-of-interest.

Processing: The 200 fMRI time courses were analyzed to extract pitch, tempo, and volume—3 important attributes for generating music. For pitch, we mapped the time course amplitudes to Musical Instrument Digital Interface (MIDI) values in the range of 36 to 84, corresponding to piano keys within a pentatonic scale. Taking advantage of these similarities, and using only rsfMRI data as input, we use basic rules of music theory to transform the data into musical form. Our project is implemented in Python using the midutil library [https://code.google.com/p/midutil/].

Results

A framework for generating music from fMRI data, based on music theory, was developed and implemented as a Python tool yielding several audio files. When listening to the results, we noticed that music differed across individual datasets. However, music generated by the same individual (4 preprocessing strategies) remained similar. Our results sound different from music obtained in a similar study using EEG and fMRI data [6].

Conclusions

In this experiment, we established a way of generating music with open fMRI data following some basic music theory principles. This resulted in a somewhat naive but pleasant musical experience. Our results also demonstrate an interesting possibility for providing feedback from fMRI activity for neurofeedback experiments.

Availability of Supporting Data

More information about this project can be found at: https://github.com/carolFrohlich/brain-orchestra

Acknowledgements

Report from 2015 Brainhack Americas (MX). The authors would like to thank the organizers and attendees of Brainhack MX.

Competing interests

None.

Author’s contributions

CF wrote the software. GD designed the functions for transforming the data to midi. DSM pick the algorithm that chooses ROIs, and CF and RCC wrote the report.

References

1. Di Martino A, Yan CG, Li Q, Denio E, Castellanos FX, Alpers K, Anderson JS, Assaf M, Bookheimer SY, Dapretto M, others. The autism brain imaging data exchange: towards a large-scale evaluation of the intrinsic brain architecture in autism. Molecular psychiatry. 2014; 19: 659–667.
2. Craddock RC, Benhajali Y, Chu C, Chournoud F, Evans A, Jakab A, Khundrakpam BS, Lewis JD, Li Q, Milham MP, Yan CG, Bellec P. The Neuro Bureau Preprocessing Initiative: open sharing of preprocessed neuroimaging data and derivatives.Frontiers in Neuroinformatics. 2013.
3. Craddock RC, Sikka S, Cheung B, Khanuja R, Ghosh SS, Yan CG, Li Q, Lurie D, Vogelstein J, Burns R, Colcombe Sj, Meneses M, Kelly C, Di Martino A, Castellanos FX, Milham M. Towards Automated Analysis of Connectomes: The Configurable Pipeline for the Analysis of Connectomes (C-PAC). Frontiers in Neuroinformatics. 2013.
4. Craddock RC, James GA, Holtzheimer PE, Hu XP, Mayberg HS. A whole brain fMRI atlas generated via spatially constrained spectral clustering. Human Brain Mapping. 2012; 33: 1914–1928.
5. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, VerHAS. Sklearn: Machine
6. Lu J, Wu D, Yang H, Luo C, Li C, Yao D. Scale-Free Brain-Wave Music from Simultaneously EEG and fMRI Recordings. PLoS ONE. 2012; 7: 1–11.

Introduction

The aim of this project was to demonstrate that an existing Matlab-based package for implementing thousands of time-series analysis methods, hctsa [https://github.com/benfulcher/hctsa] could be extended to a Python-based implementation, for potential future inclusion into Nitime [http://nipy.org/nitime/]. Recent work has contributed a comprehensive library of over 35,000 pieces of diverse time-series data, and over 7,000 unique structural features extracted from hundreds of different time-series analysis methods [1] which can be explored through an associated website [www.comp-engine.org/timeseries] and implemented using the Matlab-based code package, hctsa [https://github.com/benfulcher/hctsa]. The hctsa software provides a systematic, algorithmic platform for computing a wide range of structural properties from a single time series, including basic statistics of the distribution, linear correlation structure, stationarity, information theoretic and entropy measures, methods from the physical nonlinear time-series analysis literature, linear and nonlinear model fits, and others. Thus, hctsa can be used to map a time series to a comprehensive vector of interpretable structural features and these features can then be systematically compared to determine and understand the most useful features for a given scientific objective (e.g., features of an EEG signal that help classify different patient groups).

In order to apply highly comparative time-series analysis in the neuroscience community, it would be desirable to implement some time-series analysis methods into Nitime [http://nipy.org/nitime/], a Python-based software package for performing time-series analysis on neuroscience data. Implementation of useful time-series features into python, and potential integration with Nitime, would not only facilitate their use by the neuroscience community, but also their maintenance and development within an open source framework.

Approach

An illustration of the approach is shown in Fig. 9

Each time series is converted to a vector of thousands of informative features using the hctsa package; machine-learning methods can then be used to determine the most useful features (e.g., that best discriminate patient groups, and where in the brain the best discrimination occurs).

In this project, we wanted to demonstrate a feasible pathway for incorporating these useful features into the Nitime package.

Results

I successfully implemented a handful of basic time-series analysis functions from Matlab into python using partials (a python function that freezes a given set of input arguments to a more general function).

The proof-of-principle implementation has full support for vectors of data stored in numpy arrays, and basic support for the Nitime data format (extracting the data vector from the Nitime TimeSeries class for evenly sampled data).

Conclusions

Our results demonstrate that time-series analysis methods, discovered using the hctsa package [https://github.com/benfulcher/hctsa], can be implemented natively in python in a systematic way, with basic support for the time-series format used in Nitime.

This will help facilitate future work on time-series analysis to be incorporated straightforwardly into this open source environment. Although there are no plans to reimplement the full hctsa feature library in python, our hope is that published work describing useful time-series features (discovered using the hctsa library) can also contribute to a Python implementation, to promote its use by the neuroscience community.

Availability of supporting data

More information about this project can be found at: https://github.com/benfulcher/hctsa_python

Competing interests

None.

Author’s contributions

BF wrote the software and the report.

Acknowledgements

Report from 2015 OHBM Hackathon (HI). The authors would like to thank the organizers and attendees of the 2015 OHBM Hackathon.

References

1. Fulcher Ben D, Little Max A, Jones Nick S. Highly comparative time-series analysis: the empirical structure of time series and their methods. J Roy Soc Interface. 2013; 10: 20130048.