PyBCI: A Python Package for Brain-Computer Interface (BCI) Design

Liam Booth [1], **Aziz Asghar** [2], **and Anthony Bateson** [1]

1 Faculty of Science and Engineering, University of Hull, United Kingdom. **2** Centre for Anatomical and Human Sciences, Hull York Medical School, University of Hull, United Kingdom.

Summary:

PyBCI is an open-source Python framework designed to streamline brain-computer interface (BCI) research. It offers a comprehensive platform for real-time data acquisition, labeling, classification and analysis. PyBCI is compatible with a wide range of time-series hardware and software data sources, thanks to its integration with the Lab Streaming Layer (LSL) protocol (Kothe et al., 2023).

Statement of Need:

BCI research brings together diverse fields like neuroscience, engineering, and data science, requiring specialized tools for data acquisition, feature extraction, and real-time analysis. Existing solutions may offer partial functionalities or be cumbersome to use, slowing down the pace of innovation. PyBCI addresses these challenges by providing a flexible, Python-based platform aimed at researchers and developers in the BCI domain. Assuming a foundational understanding of Python, the software serves as a comprehensive solution for both academic and industry professionals.

Designed to be lightweight and user-friendly, PyBCI emphasizes quick customization and integrates seamlessly with the Lab Streaming Layer (LSL) for data acquisition and labeling (Kothe et al., 2023). The platform incorporates reputable machine learning libraries like PyTorch (Paszke et al., 2019), TensorFlow (Abadi et al., 2015), and Scikit-learn (Pedregosa et al., 2011), as well as feature extraction tools such as Antropy (Vallat, 2023), NumPy (Oliphant, 2006), and SciPy (Virtanen et al., 2020). This integration allows users to focus more on their research and less on software development. While a detailed comparison with other software solutions will follow in the 'State of the Field' section, PyBCI sets itself apart through its emphasis on ease of use and technological integration.

State of the Field:

There are a variety of BCI software packages available, each with its own advantages and limitations. Notable packages include solutions like OpenViBE (Renard et al., 2010) and BCI2000 (Schalk et al., 2004) that offer ease of use for those without programming expertise. BciPy (Memmott et al., 2021), another Python-based platform, provides some level of customization but does not allow for the easy integration of popular machine learning libraries. In contrast, PyBCI offers seamless integration with a variety of machine learning libraries and feature extraction tools. This flexibility makes PyBCI a robust choice for researchers seeking a tailored, code-based approach to their BCI experiments.

Software functionality and performance:

PyBCI accelerates the pace of BCI research by streamlining data collection, processing, and model analysis. It uses the Lab Streaming Layer (LSL) to handle data acquisition and labelling, allowing for real-time, synchronous data collection from multiple devices (Kothe et al., 2023). Samples are collected in chunks from the LSL data streams and stored in pre-allocated NumPy arrays. When in training mode based on a configurable time window before and after each marker type. When in test mode, data is continuously processed and analysed based on the global epoch timing settings. For feature extraction, PyBCI leverages the power of NumPy (Oliphant, 2006), SciPy (Virtanen et al., 2020), and Antropy (Vallat, 2023), robust Python libraries known for their efficiency in handling numerical operations. Machine learning, a crucial component of BCI research, is facilitated with PyTorch (Paszke et al., 2019), SciKit-learn (Pedregosa et al., 2011) and TensorFlow (Abadi et al., 2015). Scikit-learn offers a wide range of traditional algorithms for classification, including things like regression, and clustering, while TensorFlow and PyTorch provide comprehensive ecosystems for developing and training bespoke deep learning machine learning models.

Impact:

By providing a comprehensive, open-source platform for BCI research, PyBCI aims to advance the field. When integrated with off-the-shelf devices that are LSL-enabled, as well as with pre-built LSL data viewers and marker delivery systems, PyBCI facilitates the efficient design, testing, and implementation of advanced BCI experiments. The integration of LSL, PyTorch, Scikit-learn, TensorFlow, Antropy, NumPy, and SciPy into one platform simplifies the research process, encouraging innovation and collaboration in the field of brain computer/human machine interfaces.

Acknowledgements

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., … Zheng, X. (2015). *TensorFlow: Large-scale machine learning on heterogeneous systems*. https://www.tensorflow.org/

Bateson, A., & Asghar, A. (2021). Development and evaluation of a smartphone-based electroencephalography (EEG) system. *IEEE Access*, *PP*, 1–1. https://doi.org/10.1109/ACCESS.2021.3079992

Kothe, C., Stenner, T., Boulay, C., Grivich, M., Medine, D., tobiasherzke, chausner, Grimm, G., xloem, Biancarelli, A., Mansencal, B., Maanen, P., Frey, J., Chen, J., kyucrane, Powell, S., Clisson, P., & phfix. (2023). *Sccn/liblsl: v1.16.2* (Version v1.16.2). Zenodo. https://doi.org/10.5281/zenodo.7978343

Memmott, T., Koçanaoğulları, A., Lawhead, M., Klee, D., Dudy, S., Fried-Oken, M., & Oken, B. (2021). BciPy: Brain–computer interface software in Python. *Brain-Computer Interfaces*, *8*(4), 137–153. https://doi.org/10.1080/2326263X.2021.1878727

Oliphant, T. E. (2006). *A guide to NumPy* (Vol. 1). Trelgol Publishing USA.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., … Chintala, S. (2019). *PyTorch: An imperative style, high-performance deep learning library* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett, Eds.; p. 80248035). Curran Associates, Inc. Curran Associates, Inc. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Renard, Y., Lotte, F., Gibert, G., Congedo, M., Maby, E., Delannoy, V., Bertrand, O., & Lécuyer, A. (2010). OpenViBE: An open-source software platform to design, test, and use brain–computer interfaces in real and virtual environments. *Presence*, *19*(1), 35–53. https://doi.org/10.1162/pres.19.1.35

Schalk, G., McFarland, D. J., Hinterberger, T., Birbaumer, N., & Wolpaw, J. R. (2004). BCI2000: A general-purpose brain-computer interface (BCI) system. *IEEE Transactions on Biomedical Engineering*, *51*(6), 1034–1043. https://doi.org/10.1109/TBME.2004.827072

Vallat, R. (2023). AntroPy: Entropy and complexity of (EEG) time-series in Python. In *GitHub repository*. https://github.com/raphaelvallat/antropy; GitHub.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. https://doi.org/10.1038/s41592-019-0686-2