The Journal of Open Source Software

Balsa: A Fast C++ Random Forest Classifier with Commandline and Python Interface

Tobias Borsdorff [1¶], **Denis de Leeuw Duarte**[2], **Joris van Zwieten**[2], **Soumyajit Mandal** [1¶], **and Jochen Landgraf** [1]

1 SRON Netherlands Institute for Space Research, The Netherlands **2** Jigsaw B.V., The Netherlands ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

A random forest classifier is a widely used machine learning method that builds upon the strengths of decision trees (Pedregosa et al., 2011). It combines the outputs of multiple decision trees to improve predictive accuracy, reduce the risk of overfitting, and effectively identify outliers within datasets. A decision tree is a simple, intuitive model that splits input data into subsets based on the values of specific features, forming a hierarchical structure with decision nodes leading to predicted outcomes (Breiman, 2001). Balsa is a highly efficient C++ implementation of the random forest classifier concept, built with both runtime and memory performance as key design priorities. Balsa provides multi-threaded and distributed training capabilities, allowing users to scale machine learning processes across multiple computing cores or distributed systems by training separate random forests and combining them at the end. It supports both binary classification tasks as well as multi-label classification, expanding its use for a variety of machine learning challenges. Moreover, Balsa includes comprehensive tools for feature importance evaluation, prediction performance metrics, and statistical analysis, enabling users to gain insights into their data and model performance. To ensure ease of use, Balsa employs a compact and storage-efficient binary format for saving fully trained random forests. This allows models to be quickly reloaded for future predictions. Designed to fit seamlessly into existing C++ development workflows, Balsa can integrate easily into custom machine learning pipelines. Alternatively, users can employ Balsa through the command line interface or via a versatile Python interface, which can be easily installed with pip. This design provides flexibility for users working in diverse programming environments and across a wide range of machine learning use cases. Balsa performance, flexibility, and ease of integration make it a reliable choice for researchers and developers who require a fast, scalable, and efficient random forest implementation for a variety of machine learning tasks. The software package is hosted on GitHub, accompanied by a comprehensive user guide. This guide includes detailed installation instructions and multiple examples demonstrating various use cases (T. Borsdorff et al., 2024).

Statement of need

Balsa was developed by SRON and Jigsaw to support ESA's operational processing of TROPOMI CH4 satellite data (Lorente et al., 2021, 2023), specifically to identify and remove measurements contaminated by cloud interference (Tobias Borsdorff, Martinez-Velarte, et al., 2024). During the beta phase , the team utilized the excellent scikit-learn (sklearn) implementation (Pedregosa et al., 2011) of the random forest classifier concept . However, for integration into an operational framework, a C++ implementation with improved runtime and memory efficiency was required. This led to the development of Balsa, which is designed to mimic the sklearn implementation while addressing these performance demands. Balsa is now fully operational within ESA's

data processing framework (Tobias Borsdorff, Mandal, et al., 2024a, 2024b). Further planned applications include improving the posteriori quality filtering of TROPOMI data products. One key challenge involves managing multiple random forests simultaneously in memory to optimize efficiency. Moreover, Balsa will play a pivotal role in supporting the upcoming Near Real-Time TROPOMI CH4 product, which requires fast and efficient predictions to meet strict processing time constraints. Although Balsa was developed in support of TROPOMI CH4 processing, it is designed to be entirely independent of any specific application. It serves as a universal machine learning toolbox that can be applied across a variety of use cases beyond TROPOMI data processing. Its flexibility, high performance, and ease of integration make it an invaluable tool for any application requiring scalable, efficient random forest-based machine learning.

Acknowledgements

Balsa was developed for the Netherlands Institute of Space Research by Jigsaw B.V. in The Netherlands, using funding from the European Space Agency.

References

Borsdorff, T., Duarte, D. L., Zwieten, J. van, & Landgraf, J. (2024). Balsa: A fast c++ random forest classifier. In *GitHub repository*. GitHub. https://github.com/borsdorff/Balsa

Borsdorff, Tobias, Mandal, S., Martinez Velarte, M., Barr, A., & Landgraf, J. (2024a). *Algorithm theoretical baseline document for sentinel-5 precursor: TROPOMI CH4 random forest cloud filter*. https://doi.org/10.5281/zenodo.14186320

Borsdorff, Tobias, Mandal, S., Martinez Velarte, M., Barr, A., & Landgraf, J. (2024b). *Product user manual for the random forest classifier (RFC) c++ implementation to be used for TROPOMI CH4* (Version 1.0.0) [Computer software]. Zenodo. https://doi.org/10.5281/zenodo.14186406

Borsdorff, Tobias, Martinez-Velarte, M. C., Sneep, M., Linden, M. ter, & Landgraf, J. (2024). Random forest classifier for cloud clearing of the operational TROPOMI XCH4 product. *Remote Sensing*, *16*(7). https://doi.org/10.3390/rs16071208

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/A:1010933404324

Lorente, A., Borsdorff, T., Butz, A., Hasekamp, O., Brugh, J. aan de, Schneider, A., Wu, L., Hase, F., Kivi, R., Wunch, D., Pollard, D. F., Shiomi, K., Deutscher, N. M., Velazco, V. A., Roehl, C. M., Wennberg, P. O., Warneke, T., & Landgraf, J. (2021). Methane retrieved from TROPOMI: Improvement of the data product and validation of the first 2 years of measurements. *Atmospheric Measurement Techniques*, *14*(1), 665–684. https://doi.org/10.5194/amt-14-665-2021

Lorente, A., Borsdorff, T., Martinez-Velarte, M. C., & Landgraf, J. (2023). Accounting for surface reflectance spectral features in TROPOMI methane retrievals. *Atmospheric Measurement Techniques*, *16*(6), 1597–1608. https://doi.org/10.5194/amt-16-1597-2023

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.