




Salt: Multimodal Multitask Machine Learning for High Energy Physics

Jackson Barr¹, Diptaparna Biswas², Maxence Draguet³, Philipp Gadow⁴, Emil Haines¹, Osama Karkout⁵, Dmitrii Kobylanskii⁶, Wei Sheng Lai¹, Matthew Leigh⁷, Nicholas Luongo¹⁰, Ivan Oleksiyuk⁷, Nikita Pond¹, Sébastien Rettie⁴, Andrius Vaitkus¹, Samuel Van Stroud¹, and Johannes Wagner⁹

1 University College London, United Kingdom 2 University of Siegen 3 University of Oxford, United Kingdom 4 European Laboratory for Particle Physics CERN, Switzerland 5 Nikhef 6 Department of Particle Physics and Astrophysics, Weizmann Institute of Science, Israel 7 Université de Genève, Switzerland 8 Technical University of Munich, Germany 9 University of California, Berkeley 10 Argonne National Laboratory

DOI: [10.21105/joss.07217](https://doi.org/10.21105/joss.07217)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Arfon Smith](#) 

Reviewers:

- [@divijghose](#)
- [@GarrettMerz](#)

Submitted: 12 March 2024

Published: 31 August 2025

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

Summary

High energy physics studies the fundamental particles and forces that constitute the universe, often through experiments conducted in large particle accelerators such as the Large Hadron Collider (LHC) ([Evans & Bryant, 2008](#)). Salt is a Python application developed for the high energy physics community that streamlines the training and deployment of advanced machine learning (ML) models, making them more accessible and promoting shared best practices. Salt features a generic multimodal, multitask model skeleton which, coupled with a strong emphasis on modularity, configurability, and ease of use, can be used to tackle a wide variety of high energy physics ML applications.

Some key features of Salt are listed below:

- Based on established frameworks: Salt is built upon PyTorch ([Paszke et al., 2019](#)) and Lightning ([Falcon & The PyTorch Lightning team, 2019](#)) for maximum performance and scalability with minimal boilerplate code.
- Multimodal, multitask models: Salt models support multimodal inputs and can be configured to perform various tasks such as classification, regression, segmentation, and edge classification tasks. Any combination of these can be used to flexibly define models for multitask learning problems.
- Customisable and extensible: Salt supports full customisation of training and model configuration through YAML config files. Its modular design allows for the easy integration of custom dataloaders, layers, and models.
- Train at scale: Salt can handle large volumes of data with efficient HDF5 ([The HDF Group, 1997](#)) dataloaders. It also includes multi-GPU support from Lightning, enabling distributed training.
- Deployment ready: Salt facilitates ONNX ([Bai et al., 2019](#)) serialisation for integrating models into C++ based software environments.

Statement of need

In high energy physics research the reliance on ML for data analysis and object classification is growing ([Cagnotta et al., 2022](#); [Guest et al., 2018](#)). Salt meets this growing need by providing a versatile, performant, and user-friendly tool for developing advanced ML models. Salt was

originally developed to train state of the art flavour tagging models at the ATLAS experiment ([ATLAS Collaboration, 2008](#)) at the LHC. Flavour tagging, the identification of jets from bottom and charm quarks, plays a crucial role in analysing ATLAS collision data. This process is key for precision Standard Model measurements, particularly in the characterisation of the Higgs boson, and for investigating new phenomena. The unique characteristics of hadrons containing bottom and charm quarks – such as their long lifetimes, high mass, and high decay multiplicity – create distinct signatures in particle detectors that can be effectively exploited by ML algorithms. The presence of hadrons containing bottom and charm quarks can be inferred via the identification of approximately 3-5 reconstructed charged particle trajectories from the weak decay of the heavy flavour hadron amidst several more tracks from the primary proton-proton interaction vertex.

While initially developed for flavour tagging, Salt has evolved into a flexible tool that can be used for a wide range of tasks, from object and event classification, regression of object properties, to object reconstruction (via edge classification or input segmentation), demonstrating its broad applicability across various data analysis challenges in high energy physics.

Model Architecture

Salt is designed to be fully modular, but ships with a flexible model architecture that can be configured for a variety of use cases. This architecture facilitates the training of multimodal and multitask models as depicted in [Figure 1](#), and is designed to take advantage of multiple input modalities. In the context of jet classification, these input modalities might include global features of the jet and varying numbers of jet constituents such as charged particle trajectories, calorimeter energy depositions, reconstructed leptons, or inner detector spacepoints. The architecture is described briefly below. First, each input type (e.g., tracks, calorimeter deposits) is independently projected into a common embedding space of fixed dimension using separate initialisation networks. These initialisation networks can optionally concatenate global features with constituent features and apply positional encoding to certain features, for example azimuthal angle. Once embedded, the different types of constituents are considered to be in the same semantic space and are processed together by a transformer encoder that allows them to interact through stacked multi-head attention layers. The encoder maintains the same embedding dimension throughout its layers and can optionally update edge features if they are present. The encoder then outputs to a set of task-specific modules, each tailored to a specific learning objective. The initialisation networks, transformer encoder, and task-specific networks are trained together. This approach allows the model to leverage all the available detector information, leading to improved performance. Concrete examples of this architecture are in use at ATLAS ([Collaboration, 2025](#); [Graph Neural Network Jet Flavour Tagging with the ATLAS Detector, 2022](#); [Transformer Neural Networks for Identifying Boosted Higgs Bosons decaying into \$b\bar{b}\$ and \$c\bar{c}\$ in ATLAS, 2023](#)).

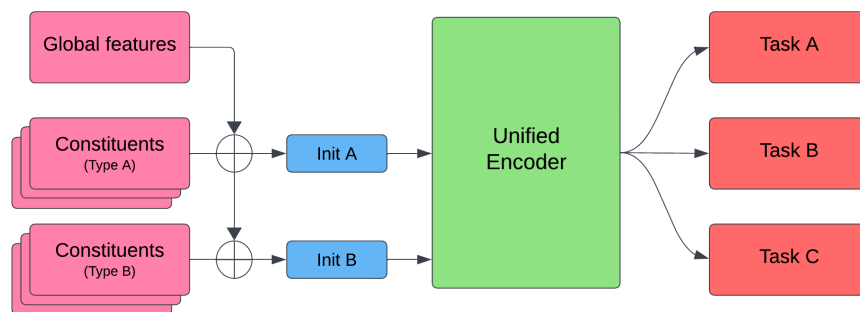


Figure 1: This diagram illustrates the flow of information within a generic model trained using Salt. In this example, global object features are provided alongside two types of constituents, “Type A” and “Type B”, which represent different input modalities such as charged particle trajectories or calorimeter energy depositions. The model is configured with three training objectives, each of which may relate to the global object or one of the constituent modalities. Concatenation is denoted by \oplus .

Related work

Umami (Barr & others, 2024) is a related software package in use at ATLAS. While Salt relies on similar preprocessing techniques as those provided by Umami, it provides several additional features which make it a more powerful and flexible tool for creating advanced ML models. Namely, Salt provides support for multimodal and multitask learning, optimised Transformer encoders (Vaswani et al., 2017), and distributed model training.

Acknowledgements

The development of Salt is part of the offline software research and development programme of the ATLAS Collaboration, and we thank the collaboration for its support and cooperation. This work is funded in part by the UK’s Science and Technology Facilities Council via University College London’s Centre for Doctoral Training in Data Intensive Science, and the Royal Society.

References

- ATLAS Collaboration. (2008). The ATLAS Experiment at the CERN Large Hadron Collider. *JINST*, 3, S08003. <https://doi.org/10.1088/1748-0221/3/08/S08003>
- Bai, J., Lu, F., Zhang, K., & others. (2019). ONNX: Open neural network exchange. In *GitHub repository*. <https://github.com/onnx/onnx>; GitHub.
- Barr, J., & others. (2024). Umami: A python toolkit for jet flavour tagging. In *GitHub repository*. <https://github.com/umami-hep/umami-preprocessing>; GitHub.
- Cagnotta, A., Carnevali, F., & Iorio, A. D. (2022). Machine learning applications for jet tagging in the CMS experiment. *Applied Sciences*, 12(20), 10574. <https://doi.org/10.3390/app122010574>
- Collaboration, A. (2025). *Transforming jet flavour tagging at ATLAS*. <https://arxiv.org/abs/2505.19689>
- Evans, L., & Bryant, P. (2008). LHC Machine. *JINST*, 3, S08001. <https://doi.org/10.1088/1748-0221/3/08/S08001>
- Falcon, W., & The PyTorch Lightning team. (2019). *PyTorch Lightning* (Version 1.4).

<https://doi.org/10.5281/zenodo.3828935>

Graph Neural Network Jet Flavour Tagging with the ATLAS Detector. (2022). CERN. <https://cds.cern.ch/record/2811135>

Guest, D., Cranmer, K., & Whiteson, D. (2018). Deep learning and its application to LHC physics. *Annual Review of Nuclear and Particle Science*, 68(1), 161–181. <https://doi.org/10.1146/annurev-nucl-101917-021019>

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. In *Advances in neural information processing systems 32* (pp. 8024–8035). Curran Associates, Inc. <http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf>

The HDF Group. (1997). *Hierarchical Data Format, version 5*.

Transformer Neural Networks for Identifying Boosted Higgs Bosons decaying into $b\bar{b}$ and $c\bar{c}$ in ATLAS. (2023). CERN. <https://cds.cern.ch/record/2866601>

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention Is All You Need. *arXiv e-Prints*, arXiv:1706.03762. <https://doi.org/10.48550/arXiv.1706.03762>