RINO: Renormalization Group Invariance with No Labels

Zichun Hao, Raghav Kansal[*,†] **Chang Sun, Maria Spiropulu**
Division of Physics, Mathematics and Astronomy
California Institute of Technology
Pasadena, CA 91125
{zhao,rkansal,chsun,smaria}@caltech.edu

Abhijith Gandrakota, Jennifer Ngadiuba
Particle Physics Division
Fermi National Accelerator Laboratory
Batavia, IL 60510
{abhijith,ngadiuba}@fnal.gov

Javier Duarte
Department of Physics
University of California San Diego
La Jolla, CA 92093
jduarte@ucsd.edu

Abstract

A common challenge with supervised machine learning (ML) in high energy physics (HEP) is the reliance on simulations for labeled data, which can often mismodel the underlying collision or detector response. To help mitigate this problem of domain shift, we propose **R**enormalization group **I**nvariance with **NO** labels (RINO), a self-supervised learning approach that can instead pretrain models directly on collision data, learning embeddings invariant to renormalization group flow scales. In this work, we pretrain a transformer-based model on jets originating from quantum chromodynamic (QCD) interactions from the JETCLASS dataset, emulating real QCD-dominated experimental data, and then finetune on the JETNET dataset — emulating simulations — for the task of identifying jets originating from top quark decays. RINO demonstrates improved generalization from the JETNET training data to JETCLASS data compared to supervised training on JETNET from scratch, demonstrating the potential for RINO pretraining on real collision data followed by fine-tuning on small, high-quality MC datasets, to improve the robustness of ML models in HEP.

1 Introduction

Machine learning (ML) applications in high-energy physics (HEP) often face the challenge of the domain shift between Monte Carlo (MC) simulations used for supervised training and real experimental data. This mismatch stems from imperfect modeling of detector effects and the underlying physical processes in simulations, and thus can require complicated calibration procedures to ensure reliable physics analyses. Meanwhile, real collision data are rarely, if ever, used to train ML models because of a lack of labels.

Self-supervised learning (SSL) offers a compelling avenue to take advantage of experimental data in ML, by training models to learn meaningful representations of the data without labels that have been shown to generalize across tasks and even domains [1–3]. We therefore explore the

[*] Also affiliated with the Fermi National Accelerator Laboratory, Batavia, IL 60510, USA
[†] Now at Bexorg, Inc.

Machine Learning and the Physical Sciences Workshop, NeurIPS 2025.

potential of SSL in exploiting the vast quantities of unlabeled collision data available for pretraining, followed by supervised fine-tuning on smaller, task-specific MC datasets. Specifically, we propose **R**enormalization group **I**nvariance with **NO** labels (RINO), an SSL technique similar to DINO [4–6], that uses self-distillation to encourage representations invariant to the energy scale of the physical process.

To test this strategy, we consider datasets of *jets*, which are collimated sprays of particles resulting from the showering and hadronization of quarks and gluon produced at high energy colliders, such as the CERN Large Hadron Collider (LHC) [7]. Jets are ubiquitous at the LHC and identifying the particles that initiate them is a critical step in data analysis. They are also high-dimensional, complex data structures — with 100s of particles per jet, each with multiple features — providing a fertile playground for ML techniques in physics. Indeed, jet classification, or "tagging", has been the focus of numerous advances in ML [8–15]. Thus, we demonstrate the effectiveness of RINO in reducing domain bias and improving the performance in real data on the downstream task of jet tagging. Our code is provided in `https://github.com/zichunhao/RINO`.

2 Related Work

SSL in HEP Recently, there has been growing interest in applying SSL techniques to HEP, including JetCLR [16, 17], masked particle modeling (MPM) [18, 19], resimulation [20], OmniJet-α [21], J-JEPA [22], and HEP-JEPA [23]. Notably, MPM explores pretraining on real collision data and fine-tuning on simulated datasets, but results show fine-tuned models do not consistently outperform fixed backbone representations, indicating domain adaptation challenges. Other methods are based on supervised learning for pretraining [24–27].

DINO Contrastive learning methods such as SimCLR [28], MoCo [29], and SwAV [30] learn representations by maximizing agreement between augmented views. DINO [4–6] extends this with teacher-student self-distillation, successfully learning attention maps in vision transformers without labels. However, standard augmentations in computer vision, such as color jittering and cropping, do not respect the inductive biases of HEP data; for example, unlike images, jet constituents have highly global correlations, with cropping thus an ill-suited augmentation. In this work, we instead design a physics-informed augmentation for jets.

3 Methodology

3.1 RINO

RINO is a self-supervised learning framework similar to DINO [4–6], where the different "views" correspond to its composition at different energy scales during its evolution. While we cannot directly access this information in data, where we see only its final form as stable hadrons in the detector, we are able to approximate it using its clustering history via the k_T algorithm [31]. The k_T clustering algorithm is an iterative jet clustering method that prioritizes the combination of objects with low transverse momentum p_T and thus mimics the reverse process of parton showering. In particular, subjets defined by the k_T algorithm have been shown to correspond well to the underlying hard decay processes [32, 33].

From a theoretical perspective, each clustering step integrates out degrees of freedom below a characteristic energy scale. Different clustering depths thus probe jets at different scales, with fewer clusters, or subjets, representing coarse-grained descriptions and more subjets preserving fine structure, effectively accessing the jet at different stages of its evolution according to the quantum chromodynamic (QCD) renormalization group flow [34]. Thus, by creating different views through varying k_T clustering steps, RINO motivates models to learn representations invariant to the energy scale, a potent inductive bias in HEP.

3.2 Training Strategy

To evaluate the efficacy of pretraining on real data in reducing domain bias, we conduct the following experiments, treating the popular JetNet [35] and JetClass [36] datasets as simulation and data,

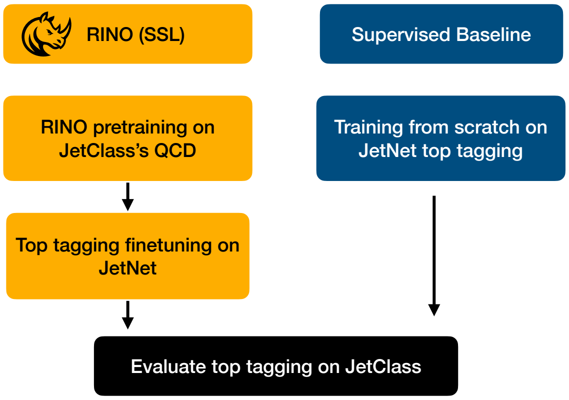
RINO (SSL)
Supervised Baseline
RINO pretraining on JetClass's QCD
Training from scratch on JetNet top tagging
Top tagging finetuning on JetNet
Evaluate top tagging on JetClass

Figure 1: RINO training strategy (left): pretrain on JETCLASS QCD jets as a proxy for real data, then finetune on JETNET top tagging. Supervised baseline strategy (right): train from scratch on JETNET. Both strategies are then evaluated on top tagging on JETCLASS.

respectively. The different simulation frameworks used in JETCLASS and JETNET allow us to meaningfully approximate the MC-to-data domain shifts commonly encountered in HEP.

Our baseline strategy is to do a supervised training entirely on JETNET for the task of discriminating between jets originating from gluons and light quarks (QCD jets) and those from top quark decays — a task referred to as "top tagging" — and evaluate its performance on JETCLASS (Fig. 1, right). This is analogous to the conventional HEP paradigm of training on simulations but applying to data.

The RINO strategy instead proposes first pretraining on data, which is dominated by QCD jets at the LHC. To emulate this, we pretrain only on QCD jets in JETCLASS, then perform a supervised *finetuning* on the smaller JETNET dataset for top tagging, and finally evaluate them on JETCLASS as well.

4 Experiments

4.1 Models and Training

We implement four transformer [37] encoder models in PyTorch [38] with different sizes and embedding dimensions: nano (32D embedding, 50k parameters), lite (64D, 200k), mini (128D, 1M), and base (256D, 5M). Full architectural details are provided in Appendix A.1.

Pretraining RINO follows the DINO framework with a teacher-student architecture processing different augmented views of the same jet. We use k_{T} clustering via FASTJET [39] to generate "global" views (corresponding to $\{1, 2, 3, 4\}$ subjets) for both networks, while the student additionally receives "local" views ($\{8, 16, 32, 64\}$ subjets, along with the original particle-level view). The teacher is updated via exponential momentum average (EMA) [40] with momentum starting at 0.992 and cosine annealing to 1.0 in the course of training.

Finetuning For the downstream task, we replace the pretraining projection head with task-specific classification heads. We explored different adaptation strategies, including linear probing [41], which freezes the backbone entirely, and joint finetuning of both head and backbone using reduced learning rate factors ($0.01\times$ and $0.1\times$) for the backbone. We observed that linear probing fails to fully leverage the representational capacity of the pretrained backbone, while aggressive finetuning often overfits to the downstream task. To address these limitations, we adopt the LP-FT (Linear Probing then Fine-Tuning) strategy [42, 43], which first trains the classification head with a frozen backbone, and then unfreezes the entire model and continues training with a reduced backbone learning rate.

We evaluate two head architectures: RINO-Linear, which employs a single linear layer directly from the class token representation to the output; and RINO-MLP, which uses a multi-layer perceptron (MLP) with GELU activation and dropout regularization. For the supervised baseline strategy, the same model architectures are trained until convergence on the JetNet dataset. Complete training details are given in Appendix A.2.

4.2 Results

Backbone Embeddings We analyze the pretrained embeddings using JetClass's hadronic top (Tbqq) class. As shown in Figure 2, both PCA and t-SNE [44] reveal good separation of top and QCD jets in the embedding space. We quantitatively assess embedding quality through k-nearest neighbor (k-NN) classification [45] with $k = 20$. The nano, lite, mini, and base models achieve accuracies of 0.866 ± 0.001, 0.859 ± 0.002, 0.860 ± 0.001, and 0.866 ± 0.002, respectively. To further validate the embedding quality, we evaluate boosted decision trees (BDT) [46] trained on the same embeddings, achieving accuracies of 0.877 ± 0.002, 0.879 ± 0.002, 0.883 ± 0.002, and 0.886 ± 0.001 for the respective model sizes. These results demonstrate that RINO produces rich, discriminative embeddings with meaningful physical information for jet classification, despite being pretrained on QCD jets alone.

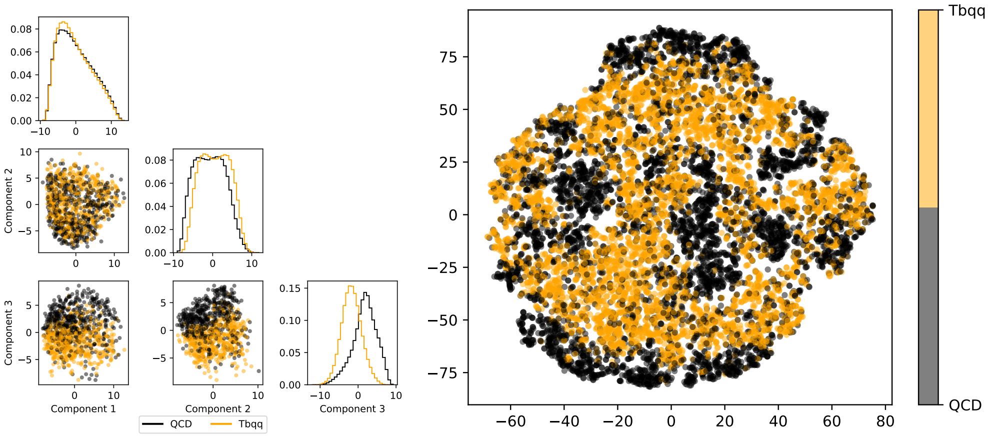

Figure 2: Visualization of learned jet representations from the base model using PCA (left) and t-SNE embedding (right). Jet representations of all five models are shown in Appendix A.3.

Top Tagging Table 1 summarizes classification performance across model architectures and training strategies on JetClass and JetNet. All results are obtained using 10-fold cross-validation with 80:20 train-test splits, where different random seeds determine the data partitioning for each fold. RINO demonstrates improvements over the supervised baseline strategy on JetClass across all model sizes, with gains ranging from 15% to 26%, demonstrating that the RINO pretraining learns transferable representations and reduces the domain bias. On JetNet, supervised baselines achieve higher accuracy as expected for in-domain evaluation, with a performance gap ranging from 4% to 5%. Nevertheless, this is a reasonable trade-off for the substantial cross-domain benefits: RINO maintains competitive performance while providing transferable representations across the two datasets.

5 Discussion and Broader Impact

We introduce RINO, a self-supervised learning framework for learning representations invariant to the energy scale of high-energy physics processes without labels. Using JetClass QCD to emulate real collision data for pretraining and then performing supervised finetuning on the JetNet dataset, emulating simulations, RINO demonstrates significantly improved generalization across domains versus supervised-only baselines, with up to 26% higher accuracy. This suggests the potential of pretraining on real experimental data followed by fine-tuning on limited simulations for specific tasks, reducing reliance on MC simulations while improving robustness. Future work will develop fine-tuning methods to better leverage the rich embeddings learned by RINO. RINO has the potential for significant broader impact in reducing biases due to simulation-only training in fundamental physics, but future work may also explore more rigorous metrics to quantify robustness.

Table 1: Classification accuracy comparison across model sizes and training strategies on JetClass and JetNet datasets. Results report $\mu \pm \sigma$ over 10-fold cross-validation with $80:20$ train-test splits.

Model size	Strategy	JetClass Accuracy	JetNet Accuracy
nano	Supervised	0.601 ± 0.060	$\mathbf{0.910 \pm 0.001}$
nano	RINO-Linear	0.748 ± 0.005	0.858 ± 0.001
nano	RINO-MLP	$\mathbf{0.755 \pm 0.012}$	0.863 ± 0.003
lite	Supervised	0.551 ± 0.038	$\mathbf{0.910 \pm 0.001}$
lite	RINO-Linear	0.810 ± 0.004	0.864 ± 0.002
lite	RINO-MLP	$\mathbf{0.812 \pm 0.005}$	0.865 ± 0.003
mini	Supervised	0.595 ± 0.049	$\mathbf{0.910 \pm 0.001}$
mini	RINO-Linear	$\mathbf{0.778 \pm 0.005}$	0.857 ± 0.002
mini	RINO-MLP	0.769 ± 0.013	0.862 ± 0.002
base	Supervised	0.629 ± 0.062	$\mathbf{0.910 \pm 0.001}$
base	RINO-Linear	$\mathbf{0.804 \pm 0.001}$	0.868 ± 0.001
base	RINO-MLP	0.772 ± 0.025	0.869 ± 0.002

Acknowledgments and Disclosure of Funding

Z.H., R.K., C.S., and M.S. are supported by the California Institute of Technology High Energy Physics under Contract DE-SC0011925 with the United States Department of Energy, Office of High Energy Physics. Z.H. wants to thank the Fermilab LPC Guests and Visitors program. A.G., J.N., and J.D. are supported by the DOE Office of Science, Award No. DE-SC0023524, FermiForward Discovery Group, LLC under Contract No. 89243024CSC000002 with the U.S. Department of Energy, Office of Science, Office of High Energy Physics, LDRD L2024-066-1, Fermilab, DOE Office of Science, Office of High Energy Physics "Designing efficient edge AI with physics phenomena" Project (DE-FOA-0002705), DOE Office of Science, Office of Advanced Scientific Computing Research under the "Real-time Data Reduction Codesign at the Extreme Edge for Science" Project (DE-FOA-0002501). J.D. is also supported by the Research Corporation for Science Advancement (RCSA) under grant #CS-CSA-2023-109, Alfred P. Sloan Foundation under grant #FG-2023-20452, U.S. Department of Energy (DOE), Office of Science, Office of High Energy Physics Early Career Research program under Award No. DE-SC0021187, and the U.S. National Science Foundation (NSF) Harnessing the Data Revolution (HDR) Institute for Accelerating AI Algorithms for Data Driven Discovery (A3D3) under Cooperative Agreement PHY-2117997. R.K. and J.N. are also supported by the AI2050 program at Schmidt Futures (Grant G-23-64934). This work was performed using the National Research Platform Nautilus HyperCluster supported by NSF awards CNS-1730158, ACI-1540112, ACI-1541349, OAC-1826967, OAC-2112167, CNS-2100237, CNS-2120019, the University of California Office of the President, and the University of California San Diego's California Institute for Telecommunications and Information Technology/Qualcomm Institute.

References

[1] D. Hendrycks, M. Mazeika, S. Kadavath, and D. Song, "Using self-supervised learning can improve model robustness and uncertainty", in *Advances in Neural Information Processing Systems*, H. Wallach et al., eds., volume 32. Curran Associates, Inc., 2019. `arXiv:1906.12340`.

[2] Y. M. Asano, C. Rupprecht, and A. Vedaldi, "A critical analysis of self-supervision, or what we can learn from a single image", in *International Conference on Learning Representations*. 2020. `arXiv:1904.13132`.

[3] J. Gui et al., "A survey on self-supervised learning: Algorithms, applications, and future trends", *IEEE Trans. Pattern Anal. Mach. Intell.* **46** (2024) 9052, `doi:10.1109/TPAMI.2024.3415112`, `arXiv:2301.05712`.

[4] M. Caron et al., "Emerging properties in self-supervised vision transformers", in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, volume abs/2104.14294, p. 9630. 2021. `arXiv:2104.14294`. `doi:10.1109/ICCV48922.2021.00951`.

[5] M. Oquab et al., “DINOv2: Learning robust visual features without supervision”, *Trans. Mach. Learn. Res.* (2024) `arXiv:2304.07193`.

[6] O. Siméoni et al., “DINOv3”, 2025. `arXiv:2508.10104`.

[7] B. Andersson, “The Lund Model”, *Nucl. Phys. A* **461** (1987) 513C, `doi:10.1016/0375-9474(87)90510-0`.

[8] P. T. Komiske, E. M. Metodiev, and J. Thaler, “Energy Flow Networks: Deep Sets for Particle Jets”, *JHEP* **01** (2019) 121, `doi:10.1007/JHEP01(2019)121`, `arXiv:1810.05165`.

[9] P. W. Battaglia et al., “Interaction networks for learning about objects, relations and physics”, in *Advances in Neural Information Processing Systems*, D. Lee et al., eds., volume 29. Curran Associates, Inc., 2016. `arXiv:1612.00222`.

[10] S. Gong et al., “An efficient lorentz equivariant graph neural network for jet tagging”, *JHEP* **07** (2022) `doi:10.1007/jhep07(2022)030`, `arXiv:2201.08187`.

[11] H. Qu, C. Li, and S. Qian, “Particle Transformer for Jet Tagging”, in *Proceedings of the 39th International Conference on Machine Learning*, K. Chaudhuri et al., eds., volume 162, p. 18281. 2022. `arXiv:2202.03772`.

[12] A. Bogatskiy et al., “Explainable equivariant neural networks for particle physics: PELICAN”, *JHEP* **03** (2024) 113, `doi:10.1007/JHEP03(2024)113`, `arXiv:2307.16506`.

[13] J. Spinner et al., “Lorentz-Equivariant Geometric Algebra Transformers for High-Energy Physics”, in *Advances in Neural Information Processing Systems*, A. Globerson et al., eds., volume 37, p. 22178. Curran Associates, Inc., 2024. `arXiv:2405.14806`.

[14] J. Brehmer et al., “A Lorentz-equivariant transformer for all of the LHC”, 2024. `arXiv:2411.00446`. Submitted to *SciPost Phys.*

[15] HEP ML Community, “A Living Review of Machine Learning for Particle Physics”. `https://iml-wg.github.io/HEPML-LivingReview/`.

[16] B. M. Dillon et al., “Symmetries, safety, and self-supervision”, *SciPost Phys.* **12** (2022) 188, `doi:10.21468/SciPostPhys.12.6.188`, `arXiv:2108.04253`.

[17] Z. Zhao et al., “Large-Scale Pretraining and Finetuning for Efficient Jet Classification in Particle Physics”, in *22nd International Workshop on Advanced Computing and Analysis Techniques in Physics Research*. 2024. `arXiv:2408.09343`.

[18] T. Golling et al., “Masked particle modeling on sets: towards self-supervised high energy physics foundation models”, *Mach. Learn. Sci. Tech.* **5** (2024) 035074, `doi:10.1088/2632-2153/ad64a8`, `arXiv:2401.13537`.

[19] M. Leigh et al., “Is tokenization needed for masked particle modeling?”, *Mach. Learn. Sci. Tech.* **6** (2025) 025075, `doi:10.1088/2632-2153/addb98`, `arXiv:2409.12589`.

[20] P. Harris et al., “Resimulation-based self-supervised learning for pretraining physics foundation models”, *Phys. Rev. D* **111** (2025) 032010, `doi:10.1103/PhysRevD.111.032010`, `arXiv:2403.07066`.

[21] J. Birk, A. Hallin, and G. Kasieczka, “OmniJet-α: the first cross-task foundation model for particle physics”, *Mach. Learn. Sci. Tech.* **5** (2024) 035031, `doi:10.1088/2632-2153/ad66ad`, `arXiv:2403.05618`.

[22] S. Katel et al., “Learning Symmetry-Independent Jet Representations via Jet-Based Joint Embedding Predictive Architecture”, in *Machine Learning and the Physical Sciences Workshop at the 38th Conference on Neural Information Processing Systems*. 2024. `arXiv:2412.05333`.

[23] J. Bardhan et al., “HEP-JEPA: A foundation model for collider physics using joint embedding predictive architecture”, 2, 2025. `arXiv:2502.03933`.

[24] C. Li et al., "Accelerating Resonance Searches via Signature-Oriented Pre-training", 2024. arXiv:2405.12972.

[25] V. Mikuni and B. Nachman, "Solving key challenges in collider physics with foundation models", *Phys. Rev. D* **111** (2025) L051504, doi:10.1103/PhysRevD.111.L051504, arXiv:2404.16091.

[26] V. Mikuni and B. Nachman, "Method to simultaneously facilitate all jet physics tasks", *Phys. Rev. D* **111** (2025), no. 5, 054015, doi:10.1103/PhysRevD.111.054015, arXiv:2502.14652.

[27] F. Mokhtar et al., "Fine-tuning machine-learned particle-flow reconstruction for new detector geometries in future colliders", *Phys. Rev. D* **111** (2025), no. 9, 092015, doi:10.1103/PhysRevD.111.092015, arXiv:2503.00131.

[28] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations", in *Proceedings of the 37th International Conference on Machine Learning*, H. D. III and A. Singh, eds., volume 119, p. 1597. 2020. arXiv:2002.05709.

[29] K. He et al., "Momentum contrast for unsupervised visual representation learning", in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, p. 9726. 2020. arXiv:1911.05722. doi:10.1109/CVPR42600.2020.00975.

[30] M. Caron et al., "Unsupervised learning of visual features by contrasting cluster assignments", in *Advances in Neural Information Processing Systems*, H. Larochelle et al., eds., volume 33, p. 9912. Curran Associates, Inc., 2020. arXiv:2006.09882.

[31] S. Catani, Y. Dokshitzer, M. Seymour, and B. Webber, "Longitudinally-invariant k_{T}-clustering algorithms for hadron-hadron collisions", *Nucl. Phys. B* **406** (1993) 187, doi:10.1016/0550-3213(93)90166-M.

[32] J. Thaler and K. Van Tilburg, "Identifying Boosted Objects with N-subjettiness", *JHEP* **03** (2011) 015, doi:10.1007/JHEP03(2011)015, arXiv:1011.2268.

[33] CMS Collaboration, "A method for correcting the substructure of multiprong jets using the Lund jet plane", 2025. arXiv:2507.07775. Submitted to *JHEP*.

[34] Z. Nagy and D. E. Soper, "Multivariable evolution in final state parton shower algorithms", *Phys. Rev. D* **105** (2022) 054012, doi:10.1103/PhysRevD.105.054012, arXiv:2201.08056.

[35] R. Kansal et al., "Particle cloud generation with message passing generative adversarial networks", in *Advances in Neural Information Processing Systems*, M. Ranzato et al., eds., volume 34, p. 23858. Curran Associates, Inc., 2021. arXiv:2106.11535.

[36] H. Qu, C. Li, and S. Qian, "JetClass: A large-scale dataset for deep learning in jet physics", 2022. doi:10.5281/zenodo.6619768.

[37] A. Vaswani et al., "Attention is all you need", in *Advances in Neural Information Processing Systems*, I. Guyon et al., eds., volume 30. Curran Associates, Inc., 2017. arXiv:1706.03762.

[38] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library", in *Advances in Neural Information Processing Systems*, H. Wallach et al., eds., volume 32. Curran Associates, Inc., 2019. arXiv:1912.01703.

[39] M. Cacciari, G. P. Salam, and G. Soyez, "FastJet User Manual", *Eur. Phys. J. C* **72** (2012) 1896, doi:10.1140/epjc/s10052-012-1896-2, arXiv:1111.6097.

[40] C. C. Holt, "Forecasting seasonals and trends by exponentially weighted moving averages", *Int. J. Forecast.* **20** (2004) 5, doi:10.1016/j.ijforecast.2003.09.015.

[41] G. Alain and Y. Bengio, "Understanding intermediate layers using linear classifier probes", in *International Conference on Learning Representations*. 2017. arXiv:1610.01644.

[42] A. Kumar et al., “Fine-tuning can distort pretrained features and underperform out-of-distribution”, in *International Conference on Learning Representations*. 2022. arXiv:2202.10054.

[43] Y. Ren, S. Guo, W. Bae, and D. J. Sutherland, “How to prepare your task head for finetuning”, in *International Conference on Learning Representations*. 2023. arXiv:2302.05779.

[44] L. van der Maaten and G. Hinton, “Visualizing data using t-SNE”, *J. Mach. Learn. Res.* **9** (2008) 2579.

[45] E. Fix and J. L. Hodges, “Discriminatory analysis. nonparametric discrimination: Consistency properties”, *Int. Stat. Rev.* **57** (1989) 238, doi:10.2307/1403797.

[46] J. H. Friedman, “Greedy function approximation: A gradient boosting machine”, *Ann. Stat.* **29** (2001) 1189.

[47] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding”, 2019. https://arxiv.org/abs/1810.04805.

[48] A. Dosovitskiy et al., “An image is worth 16x16 words: Transformers for image recognition at scale”, 2021. https://arxiv.org/abs/2010.11929.

[49] M. Caron et al., “Unsupervised learning of visual features by contrasting cluster assignments”, in *Advances in Neural Information Processing Systems*, H. Larochelle et al., eds., volume 33, p. 9912. Curran Associates, Inc., 2020.

[50] A. Sablayrolles, M. Douze, C. Schmid, and H. Jégou, “Spreading vectors for similarity search”, in *International Conference on Learning Representations*. 2019. arXiv:1806.03198.

[51] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization”, 2019. https://openreview.net/forum?id=Bkg6RiCqY7.

[52] D. Weitzel et al., “The National Research Platform: Stretched, multi-tenant, scientific Kubernetes cluster”, in *Practice and Experience in Advanced Research Computing 2025: The Power of Collaboration*. Association for Computing Machinery, 2025. doi:10.1145/3708035.3736060.

[53] S. Gugger et al., “Accelerate: Training and inference at scale made simple, efficient and adaptable.”. https://github.com/huggingface/accelerate, 2022.

[54] F. Pedregosa et al., “Scikit-learn: Machine learning in Python”, *J. Mach. Learn. Res.* **12** (2011) 2825.

[55] J. D. Hunter, “Matplotlib: A 2D graphics environment”, *Comput. Sci. Eng.* **9** (2007) 90, doi:10.1109/MCSE.2007.55.

[56] C. R. Harris et al., “Array programming with NumPy”, *Nature* **585** (2020), no. 7825, 357, doi:10.1038/s41586-020-2649-2, arXiv:2006.10256.

A Technical Appendices and Supplementary Material

The code is provided in https://github.com/zichunhao/RINO.

A.1 Model Architectures

Transformer Backbones All models employ transformer encoder architecture with GELU activation functions, pre-layer normalization, and jet-initialized class tokens that integrate jet kinematics for class token initialization with particle kinematics preprocessed following the particle transformer methodology [11]. Inspired by BERT [47] and ViT [48], we take the embedding that corresponds to the jet token as the jet embedding, as shown in Figure 3. The inputs are normalized manually for training stability. The transformer encoder architecture is implemented in four configurations with systematic scaling across embedding dimensions, attention heads, layers, and feedforward dimensions:

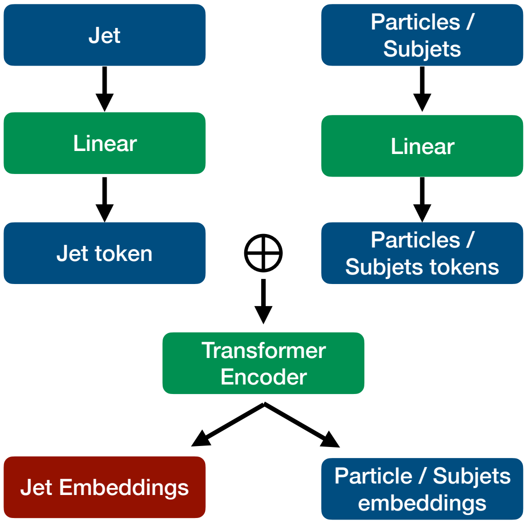

Figure 3: Model architecture of the transformer encoder backbone. The jet's representation is taken as the transformer embedding corresponding to the jet token.

- **Nano**: 32-dimensional encoder embeddings, 4 attention heads, 4 layers, 128-dimensional feedforward network (51,424 parameters)
- **Lite**: 64-dimensional encoder embeddings, 4 attention heads, 4 layers, 256-dimensional feedforward network (201,152 parameters)
- **Mini**: 128-dimensional encoder embeddings, 8 attention heads, 6 layers, 512-dimensional feedforward network (1,192,064 parameters)
- **Base**: 256-dimensional encoder embeddings, 8 attention heads, 6 layers, 1024-dimensional feedforward network (4,743,424 parameters)

DINO Projection Heads The DINO projection heads utilize multi-layer architectures with GELU activation and model-specific configurations:

- **Nano**: 128-dimensional hidden layer producing 16-dimensional representations (6,272 parameters)
- **Lite**: 256-dimensional hidden layer producing 32-dimensional representations (24,832 parameters)
- **Mini**: 512-dimensional hidden layer producing 64-dimensional representations (98,816 parameters)
- **Base**: 1024-dimensional hidden layer producing 128-dimensional representations (394,240 parameters)

All projection heads incorporate L_2 normalization prior to the final projection layer and apply weight normalization to the output layer for training stabilization and enhanced representation quality.

Classification Heads For the RINO-Linear approach, we attach a linear head to the backbone. The numbers of parameters are 16, 32, 64, 128, and 256 parameters for nano, lite, mini, and base models respectively. For the RINO-MLP approach and supervised baselines, we employ two-layer multilayer perceptrons with GELU activation and dropout regularization. The architectures scale proportionally with model size:

- **Nano**: First hidden layer of 8 dimensions, second hidden layer of 4 dimensions, dropout rate of 0.1 (304 parameters)
- **Lite**: First hidden layer of 16 dimensions, second hidden layer of 8 dimensions, dropout rate of 0.1 (1,184 parameters)

- **Mini**: First hidden layer of 32 dimensions, second hidden layer of 16 dimensions, dropout rate of 0.1 (4,672 parameters)
- **Base**: First hidden layer of 64 dimensions, second hidden layer of 32 dimensions, dropout rate of 0.1 (18,560 parameters)

A.2 Training Details

RINO pretraining Global views use $\{1, 2, 3, 4\}$-cluster configurations, while student local views use $\{8, 16, 32, 64\}$-cluster configurations plus the original particle-level view. Teacher momentum starts at 0.992 with cosine annealing to 1.0. We use a teacher temperature of 0.07 with cosine warmup from 0.04 over 20 epochs, a fixed student temperature of 0.10. The Sinkhorn-Knopp algorithm [49] is used for centering, and the KoLeo regularization [50] is added with a weight of 1.0 to prevent mode collapsing. Training employs the AdamW optimizer [51] with a learning rate of 1×10^{-4} and weight decay of 0.01, following cosine annealing over 100 epochs. The base model uses 4 A100 GPUs, whereas the other four (nano, lite, and mini) models use 5 A10/3090 GPUs at the National Research Platform [52]. Multi-GPU training is achieved by HUGGINGFACE ACCELERATE [53]. Each model takes 2-7 days to pretrain on the QCD jets from JETCLASS.

RINO Fine-Tuning A task-specific classification head (linear layer for RINO-Linear or MLP for RINO-MLP) is attached to the pretrained backbone. Training employs the AdamW optimizer with weight decay of 0.01 and an initial head learning rate of 1×10^{-3}. The backbone is frozen for the first 40 epochs, then unfrozen with a reduced learning rate of 1×10^{-5} for the nano and lite models and 1×10^{-6} for the mini and base models. The learning rate schedule follows a two-stage cosine strategy: 10-epoch warmup with learning factors scaling from 1×10^{-4} to 1.0, followed by 90-epoch cosine annealing from 1.0 to 1×10^{-3}. Binary cross-entropy loss is used with positive class weighting of 2.0 to address class imbalance. RINO-Linear models require 2 A10/3090 GPUs while RINO-MLP models require 4 A10/3090 GPUs, with training completing in under 60 minutes per model.

Supervised Baselines Training is performed from scratch using the AdamW optimizer with a learning rate of 1×10^{-4} and weight decay of 0.01, following cosine annealing for up to 1000 epochs with early stopping patience of 30 epochs. All models use 2 A10/3090 GPUs. Each model takes less than 90 minutes to train.

A.3 Experiments

Backbone Embeddings Figure 4 shows the learned jet embeddings from pretrained nano, lite, mini, and base models using PCA decomposition and t-SNE embedding. Figure 5 presents the confusion matrices of BDT classifiers for hadronic top class vs QCD class trained on these learned embeddings, which demonstrates that all models achieve well-balanced classification performance, with the base and mini models showing slightly better true positive rates for top jet identification compared to the nano and lite models. For t-SNE visualization, 10,000 randomly chosen jets are embedded and plotted. For PCA corner plots, principal component analysis is performed on the entire test dataset, with 1,000 randomly selected jets used to create the scatter plots for off-diagonal elements. PCA and t-SNE are implemented using the SCIKIT-LEARN package [54], with visualizations created using the MATPLOTLIB package [55] and array processing performed using the NUMPY package [56]. The similarity of the PCA decompositions between the mini and base models explains their similar BDT accuracies.

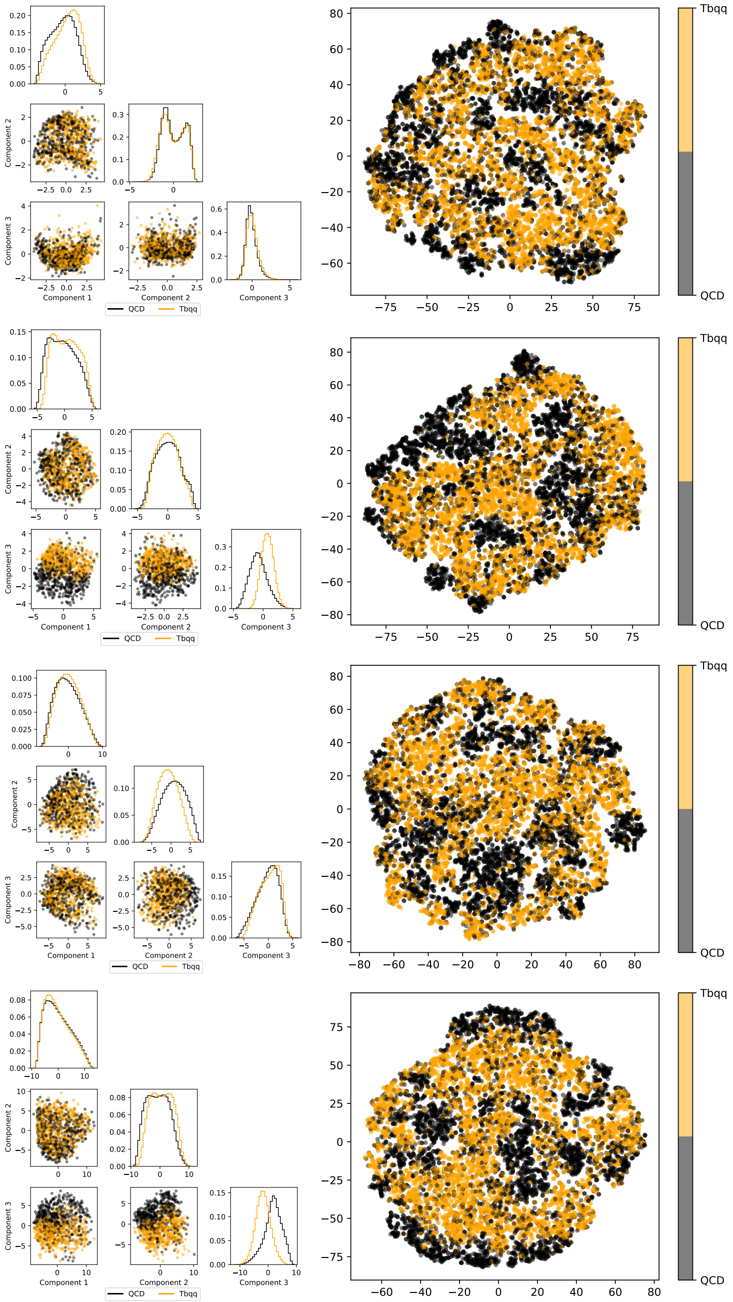

Figure 4: Visualization of learned jet representations from the nano (row 1), lite (row 2), mini (row 3), and base (row 4) models using PCA (left) and t-SNE embedding (right).

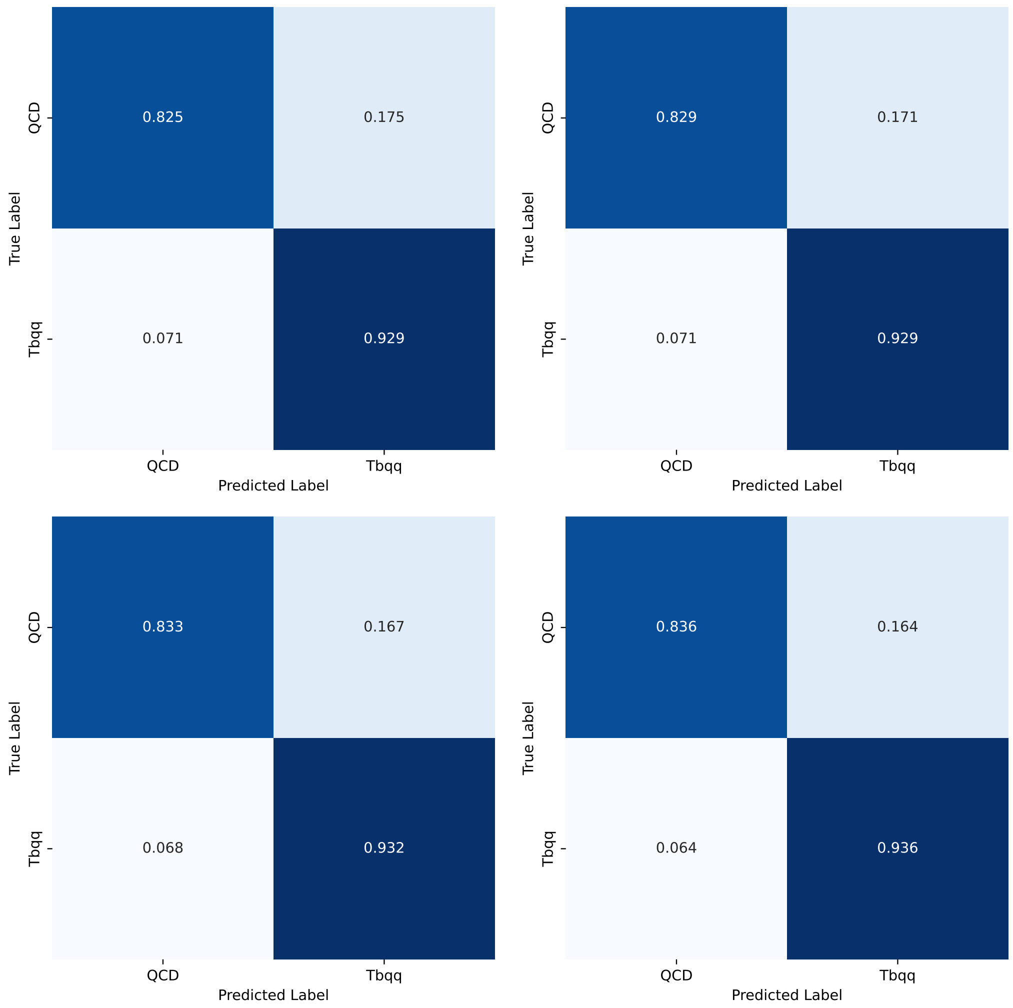

Figure 5: Confusion matrix of BDT for hadronic top class (Tbqq) vs QCD class from JetClass on the learned jet embeddings from pretrained nano (top left), lite (top right), mini (bottom left), and base (bottom right) models.