Quantum Kolmogorov-Arnold Networks for High Energy Physics Analysis at the LHC

Contact Information:

Full Name: Atharv Vaibhav Munot
Preferred Name: Atharv
Email ID: munot.atharv@gmail.com
GitHub ID: atharvmunot004

I. Synopsis

With the advent of the High-Luminosity LHC (HL-LHC), the data deluge in High Energy Physics (HEP) creates significant computational and analytical challenges. This project explores the frontier of Quantum Machine Learning (QML) by developing a quantum diffusion model using the Pennylane framework. Unlike classical deep networks, quantum diffusion models promise improved sampling complexity and representational power, which are vital for distinguishing rare physics signals amidst immense background noise.

Building on prior work with advanced function-approximator architectures like Kolmogorov–Arnold Networks (KANs), this project brings a unique perspective to designing efficient, and interpretable models. By benchmarking performance on a representative HEP dataset and comparing it to classical models (including diffusion and KAN-based networks), the project aims to quantify the tangible benefits and limitations of QML in physics workflows.

This work contributes to ML4SCI's long-term vision by laying the groundwork for practical, open-source quantum-classical hybrid analysis pipelines and by identifying opportunities for quantum advantage in scientific machine learning.

II. Benefits to the Community

This project aligns with open-source and HEP community objectives in multiple ways:

- **Novel Architecture**: KANs may provide more interpretable and efficient representations than black-box deep networks, directly aiding analysis in HEP.
- **Open-Source Tools**: The project will produce code, tutorials, and benchmarks for integrating KANs into HEP workflows, useful for researchers beyond the LHC community.
- **Cross-Disciplinary Innovation**: Demonstrating the use of KANs in HEP bridges the gap between theoretical AI/ML advancements and experimental physics.

- **Scalability**: The study will highlight trade-offs in using KANs for large-scale, high-dimensional scientific data.

III. Deliverables

- Trained quantum diffusion model.
- Benchmark of the performance on a HEP dataset compared against a classical reference model.
- Documentation and tutorial for future use.

IV. Timeline

Community Bonding Period (8th May 25 – 1st June 25)

- Familiarization with HEP workflows, datasets, and expectations.
- Finalize dataset and task (classification/anomaly detection) with mentors.
- Review foundational material on KANs and functional approximators.
- Review existing implementations of KANs (e.g., from GitHub/KAN paper).

Phase 1: Research and Prototyping (2nd June 25 – 30th June 25)

- Build and train KANs on synthetic and toy datasets.
- Select baseline classical models (e.g., MLP, XGBoost).
- **Deliverable**: Working prototype of KAN for toy data and working ML pipeline setup for HEP dataset.

Phase 2: Dataset Integration & Comparative Study (1st July 25 – 31st July 25)

- Integrate HEP dataset (from CERN Open Data or a benchmark challenge).
- Train KAN and classical models on signal vs background classification.
- Log training time, accuracy, AUC, and other metrics.
- **Deliverable**: Trained KAN model on real HEP data with comparative benchmark plots.

Phase 3: Interpretation, Documentation, and Reporting (1st Aug. 25 – 19th Aug. 25)

- Visualize learned functions and analyse interpretability.
- Evaluate generalization and overfitting tendencies.
- Write reproducible scripts, documentation, and technical report.

A buffer of one week is kept for any unprecedented delay.

V. Related Work

While most ML applications in HEP focus on traditional deep networks or tree-based models, using function-based approximators like KANs remains unexplored. Projects like CERN's ML4HEP and efforts by IRIS-HEP provide valuable infrastructure but rely on standard architectures. KANs provide a complementary approach, potentially yielding models with better generalization and interpretability. This work aims to fill the gap by integrating KANs into standard HEP workflows and evaluating their real-world performance.

Current work in the field:

Explainable AI for High Energy Physics

KAN we improve on HEP classification tasks? Kolmogorov-Arnold Networks applied to an LHC physics example

QKAN: Quantum Kolmogorov-Arnold Networks

VI. Biographical Information

I am an undergraduate studying Computer Science Engineering at COEP Technological University, Pune; with a strong foundation in machine learning, physics, and mathematical modeling. I have recently built and experimented with Kolmogorov–Arnold Networks (KANs), focusing on their mathematical grounding and practical implementation using PyTorch. This exploration led to successful approximation of complex functions in low-dimensional tasks, which I am now excited to apply in real-world scientific domains like High Energy Physics.

Beyond KANs, I've contributed to systems-level projects such as implementing custom memory allocators (SLAB and SLOB) in the xv6 operating system kernel. This work involved modifying low-level C code, analyzing memory footprint, and improving allocation efficiency, reinforcing my problem-solving and debugging skills in performance-critical environments. These experiences have honed my ability to work with large, unfamiliar codebases—skills that will be directly valuable in this GSoC project when integrating novel architectures into established HEP workflows.

Additionally, I have developed several ML models from scratch for educational and research projects and have experience working with structured scientific data using tools like NumPy, pandas, and matplotlib.

Relevant Skills:

- Python, PyTorch, NumPy, pandas
- Kolmogorov–Arnold Networks and functional approximation
- High Energy Physics data tools (uproot, awkward-array)
- Machine learning model development and benchmarking
- Systems programming, Linux kernel internals (xv6, memory allocators)
- Reproducible research practices and technical documentation

VII. Commitment & Availability

I am committed to dedicate **40 hours per week** to GSoC.

During the GSoC timeline:

- I will be available for regular communication through GitHub, email, and preferred chat platforms as per ML4SCI's practices.
- I understand the importance of timely updates and collaboration and am committed to proactively seeking help and feedback when needed.