Quantum Kolmogorov–Arnold Networks for LHC Rare-Event Classification

Name: Zih-Chen, Ciou (Jenson)
Email: jensonchiu000@gmail.com / jcciou@gapp.nthu.edu.tw
School/University: National Tsing Hua University, Taiwan
Graduation Date: June, 2026 (expected)
Major/Focus: Physics (Quantum & Photonics) / Electrical Engineering
Location: Taiwan (UTC+8)

Contents

Personal Information

- **Name:** Zih-Chen, Ciou (Jenson)
- **Email:** `jensonchiu000@gmail.com` / `jcciou@gapp.nthu.edu.tw`
- **School/University:** National Tsing Hua University, Taiwan
- **Graduation Date:** June, 2026 (expected)
- **Major/Focus:** Physics (Quantum & Photonics) / Electrical Engineering
- **Location:** Taiwan (UTC+8)

Project Repository: For transparency and community sharing, a partial upload of my proof-of-concept experiments and preliminary QKAN testing scripts is available on GitHub: https://github.com/jcciou/ml4sci$_q$*mlhep* − *testing*/*tree*/*main*

1 Motivation and Personal Expectation

I am a third-year undergraduate student majoring in Physics at National Tsing Hua University in Taiwan, with a strong academic focus on quantum information and a growing passion for applying machine learning to scientific research. My current research projects surround machine learning applications and photonic hardware design on quantum signal processing. Earlier this year, I was invited to the AAAI 2025 Conference Workshop of anomaly detection as a speaker, where I had the opportunity to attend Professor Konstantin Matchev's talks on the outcomes of last year's GSoC QML projects in the QML workshop. His introduction deeply resonated with my interests—especially in the intersection of quantum computing and high energy physics—and inspired me to get involved. Through this GSoC project, I hope to contribute to advancing Quantum Machine Learning tools such as QKAN within the Pennylane framework, and explore how they can enhance rare event detection in LHC data analysis. I am especially motivated by the long-term vision of building scalable, hybrid quantum-classical solutions for the HL-LHC era.

2 Introduction

The High Luminosity Large Hadron Collider (HL-LHC) program will generate an unprecedented amount of collision data in the coming decades, posing a significant challenge to current computational infrastructures. A central problem in such analyses is the identification of extremely rare physics signals (e.g., signatures of new particles or exotic processes) amid an immense background of Standard Model processes. Traditional workflows, while robust, may be computationally insufficient and rely on carefully tuned event selection strategies. Recent years have shown that *machine learning* (ML), especially deep neural networks, can substantially enhance the sensitivity to these rare signals by reducing backgrounds and extracting subtle correlations from high-dimensional feature spaces. However, with collisions reaching tens of billions of events, purely classical ML solutions will likely strain available HPC (High-Performance Computing) resources. This has spurred interest in investigating *quantum machine learning* (QML) as a potential next-generation approach to further speed up or improve classification tasks in high energy physics (HEP).

Why LHC HEP Analysis Requires (Quantum) ML for Rare Signal Detection

Mathematically, a typical HEP classification challenge can be phrased as finding a function

$$f(\mathbf{x}) : \mathbb{R}^n \ \rightarrow \ [0, 1],$$

where each event $\mathbf{x} \in \mathbb{R}^n$ captures (potentially hundreds of) variables—kinematic properties, detector signals, jet momenta, etc.—and $f(\mathbf{x})$ returns the probability of the event belonging to a rare signal rather than a large background. As the HL-LHC runs at higher luminosities, both n and the dataset size N grow rapidly, requiring more efficient or more powerful learning methods to exploit the data.

In classical deep learning, such as multi-layer perceptrons (MLPs), one typically uses a pipeline:

1. Linear transformations $W\mathbf{x} + \mathbf{b}$,
2. Fixed activation functions $\sigma(\cdot)$ (e.g., ReLU),
3. Summation or further transformations in deeper layers.

These MLP-based algorithms have proven extremely successful in many data-rich tasks. However, advanced HEP analyses that target rare signals—like single-top plus Higgs processes or dark matter searches—may require extremely fine-grained resolution of the input feature space. This heightened complexity leads to large networks with many parameters, adding interpretability challenges and heavy computational loads.

Quantum computing aims to further push the boundary by encoding these large feature spaces or correlation structures into the exponentially large state space of qubits. If data $\mathbf{x}$ can be embedded into a quantum state (for instance, via amplitude encoding or block-encoding), certain transformations might be more efficiently performed on quantum hardware. The ultimate hope is some form of quantum advantage—reducing training time, inference time, or capturing subtle entanglements in $\mathbf{x}$.

Potential Quantum Supremacy in Rare-Signal Classification

In the quantum machine learning paradigm, an algorithm might realize transformations $U(\theta)$ within a Hilbert space, where θ represent trainable parameters. The theoretical promise is that certain high-dimensional manipulations—like large matrix products or partial inverses (e.g., in quantum linear system solvers)—scale better in quantum settings under certain assumptions. For instance, block-encoding and quantum singular value transformation (QSVT) can embed large weight matrices or implement polynomial transformations more efficiently than classical hardware would allow, given an oracle or well-structured input. In the context of HEP, one might hypothesize that computing large sums over final states or generating amplitude-level interference patterns (akin to how jets might combine) could benefit from quantum strategies.

Indeed, references on *quantum Kolmogorov-Arnold Networks (QKAN)* [1, 2] highlight how Chebyshev expansions and polynomial transformations can be performed with quantum subroutines in $\widetilde{O}(N)$ time complexity, thereby providing a potential speedup when the dimension N of input data is large. The QKAN approach uses trainable activation functions parameterized by polynomials (like Chebyshev) but realized via quantum circuits, possibly yielding novel ways to handle extremely big data if read-in overhead is surmountable.

Nevertheless, true "quantum supremacy" is still not guaranteed in real HEP classification tasks: the practical advantage depends on overheads (e.g., data loading to qubits, error rates, circuit depth). However, ongoing research [1, 3, 2] suggests that quantum-based ML subroutines can become a strong candidate for next-generation LHC analytics once fault-tolerant or advanced NISQ hardware is available.

KAN's Main Attributes for HEP and Recent Directions

Kolmogorov-Arnold Networks (KAN) [4, 3, 5] are motivated by the Kolmogorov-Arnold representation theorem, which states that any continuous function $f(x_1, \ldots, x_n)$ can be expressed as sums and compositions of univariate functions:

$$f(\mathbf{x}) \;=\; \sum_{q=1}^{2n+1} \Phi_q\Big(\sum_{p=1}^{n} \varphi_{q,p}(x_p)\Big).$$

KAN "lifts" this from a theoretical formula into a neural-like architecture. Instead of conventional MLP edges with fixed activation, each edge in KAN carries a learnable single-variable spline (or polynomial, or sinusoidal function). Summation happens at the nodes, making the entire feedforward process a composition of 1D transformations plus additions.

For classification tasks, especially in rare-signal detection, two beneficial KAN features often stand out:

1. **Expressive and modular approximation**: Because KAN directly learns univariate transformations $\varphi_{q,p}$, it can approximate complicated local behaviors while having only moderate widths. Some tasks in HEP (like specific kinematic transforms or partially factorized variables) naturally align with KAN's local function expansions [5].

2. **Interpretability**: Each learnable activation can be plotted as a 1D function, which can be pruned or examined for physically meaningful patterns (e.g. log-likelihood-like shapes in certain kinematic variables [4]). This is in contrast to large MLP weight matrices, which can be less intuitive to interpret.

Recent efforts have extended KAN in multiple directions. For instance, *SineKAN* [3] replaces B-spline expansions with sinusoidal expansions, showing faster inference times; others have used wavelet expansions or block-encodings of polynomials to handle more complicated tasks [1, 2]. In HEP classification, preliminary studies [4] found that shallow KANs matched or slightly underperformed MLPs in raw accuracy but offered more direct interpretability and more efficient param usage once pruned.

Known Limitations of QKAN for HEP

While QKAN attempts to merge quantum hardware advantages with KAN's compositional structure, certain obstacles remain:

Quantum Data Loading: For block-encoding or amplitude encoding to provide a speedup, the mapping $\mathbf{x} \mapsto |\mathbf{x}\rangle$ must be done efficiently. However, typical HEP events can have $\mathcal{O}(100)$ or more features—still feasible for NISQ era but challenging.

Circuit Depth and Overheads: Realizing repeated polynomial expansions or multiple QKAN layers requires repeated QSVT blocks. As the layer count grows, the required circuit depth may become too large for NISQ devices, introducing errors beyond feasible correction.

No Guaranteed Super-Polynomial Gain: Although QKAN can theoretically exploit quantum subroutines, it does not automatically yield super-polynomial advantage. The advantage might be overshadowed by classical pre-processing or the need for repeated measurement.

Hyperparameter Sensitivity: Implementation details—choice of polynomial degree d, stepwise domain expansions, controlling numerical stability—can heavily affect QKAN training. Tuning them for large LHC data sets remains non-trivial.

To date, only small-scale demonstrations [1, 2] exist that tackle simplified data sets or moderate dimension. While the results look promising in controlled scenarios (e.g. amplitude-like encodings, or artificially structured classification tasks), robust testing on real LHC event datasets is still ongoing. Overcoming these flaws—especially data-upload overhead and high circuit depth—will be crucial before a QKAN-based pipeline can realistically compete with or surpass classical HPC methods in large-scale HEP analyses.

3 Proposed Methods

Our approach combines quantum-enhanced neural architectures (QKAN) with classical post-processing to detect rare signals in LHC Monte Carlo or open-data samples. We divide the methodology into four main tasks, summarized below.

3.1 Data Preprocessing and Feature Engineering

- **Dataset Selection:** We plan to use a representative LHC Monte Carlo sample (e.g., Higgs/SUSY signals) or publicly available ATLAS/CMS open data. These events typically come with multiple kinematic variables (jets, leptons, missing energy, etc.).
- **Normalization and Dimensionality Reduction:** We will first apply standard preprocessing (mean-std or min-max normalization) and optionally explore PCA or UMAP to see if an initial classical dimension reduction eases embedding into quantum states.
- **Label Definition:** We define "rare signals" vs. "background" based on event generator truth labels, ensuring enough training samples to evaluate classification performance.

3.2 Quantum Circuit Design (QKAN Layer)

- **Block-encoding + Polynomial Activation:** Following [1, 2], we implement a QKAN layer in PennyLane that uses Chebyshev or sine polynomial expansions as activation functions. We incorporate Linear Combination of Unitaries (LCU) techniques to perform weighted sums of multiple polynomial branches.
- **Circuit Depth & Resource Estimates:** For a given number of qubits (n plus auxiliary), polynomial degree (d), and layer depth (L), we estimate circuit size and error rates. This helps us decide how many layers/activations are feasible under near-term NISQ constraints.
- **Sine vs. Chebyshev Activations:** We will compare performance of sinusoidal expansions (SineKAN [3]) and Chebyshev expansions in terms of expressivity, numerical stability, and hardware gate counts.

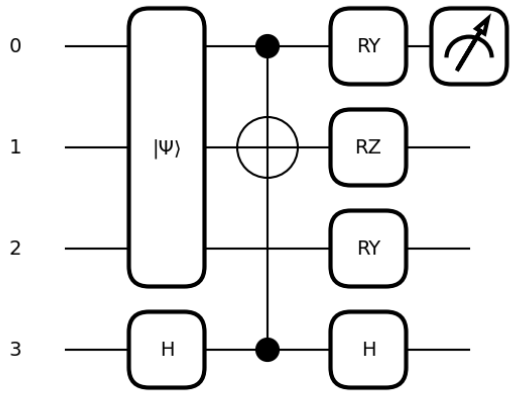

Figure 1: A schematic QKAN-style circuit with 3 data qubits (0,1,2) plus 1 ancilla (3). The block $|\Psi\rangle$ represents amplitude or block-encoding of input data, followed by controlled operations (circles/vertical lines), polynomial gates (RY, RZ), and ancilla manipulations (Hadamard). Finally, an expectation value measurement is performed on qubit 0.

3.3 Hybrid Variational Training Strategy

- **Quantum-Classical Optimization:** Using PennyLane's parameter-shift rule, we optimize both quantum parameters (e.g. polynomial coefficients in QKAN) and any classical part (e.g. final fully-connected DNN) end-to-end.
- **Mixed Architecture Exploration:** We will try at least two pipelines: (1) QKAN front-layers + classical DNN back-layers, and (2) classical front-layers + a QKAN output stage. We will measure trade-offs in training stability, speed, and interpretability.
- **Calibration and Post-Processing:** As in Section III (Hybrid Decision Pipeline), we adopt small classical classifiers (Logistic Regression, Random Forest) with Platt Scaling to ensure calibrated output probabilities for rare-event classification.

3.4 Simulation and Potential Real-Hardware Testing

- **Local Simulation:** We run detailed experiments in PennyLane's built-in simulators and Qiskit Aer, verifying performance and resilience to noise.
- **Limited NISQ Deployment:** If resources permit, we attempt small-scale QKAN circuits on real hardware (e.g. IBM Quantum). We measure training convergence and noise tolerance for a reduced (sparse) version of the QKAN layer.
- **Extension: TKAN for Temporal Data:** For multi-time-slice events or sequential subdetector signals, we aim to explore T-KAN [6] with LSTM-like gating in future expansions of this framework.

Some proof of concepts of the proposed methods with QKAN testing scripts is available on GitHub:
https://github.com/jcciou/ml4sci$_q mlhep - testing/tree/main$

4 Work Load and Time Line

We align with the GSoC 2025 schedule under a Medium-sized project (175 hours). During July 22–September 15, our availability is limited to weekends due to military service, but we will maximize coding progress in earlier and final stages.

May 8 – June 1: Community Bonding

- Study codebase for QKAN and prior Kolmogorov–Arnold expansions.
- Finalize dataset choice (e.g., Higgs signal vs. background).
- Set up environment (PennyLane, Qiskit Aer), gather HPC or GPU resources.
- Refine project scope and deliverable plan with mentors.

June 2 – July 21: Coding Phase I (Full Time)

- **Data Preprocessing & Explorations:** Standardize features, try PCA/UMAP, define an initial embedding approach.
- **QKAN Layer Implementation:** Complete block-encoding + polynomial activation in PennyLane. Validate Chebyshev vs. sine expansions on toy functions.
- **Initial Benchmarks:** Train with small classical back-end (Logistic Regression, Random Forest). Evaluate accuracy, circuit depth, run-time.
- Midterm Report submission (around July 14).

July 22 – September 15: Coding Phase II (Weekend Availability)

- **Hybrid Training & Calibration:** Use parameter-shift rule to optimize QKAN + DNN, calibrate outputs with Platt/Isotonic.
- **Noise & Resource Study:** Evaluate gate counts, error accumulation, and possible NISQ real-device test for a small QKAN circuit.
- Incremental code refactoring and documentation.

September 16 – September 30: Final Polish and Submission

- Compare final QKAN pipeline with baseline MLP or other quantum ML approaches.
- Visualize results (ROC curves, confusion matrices, 2D embeddings).
- Submit final GSoC deliverables, including code, documentation, and results.
- Outline potential next-step: T-KAN for multi-time-slice data.

5 Expected Results

1. Effective QKAN Embedding:

- Confirm block-encoding + polynomial expansions can embed LHC data in a latent space suitable for separating rare signals.
- Show improvements (e.g. accuracy, F1-score) over baseline classical MLPs under similar parameter budgets.

2. Interpretability and Calibration:

- Logistic/Random Forest back-end identifies which latent-space features are most relevant, providing insight into the physics.
- Platt/Isotonic scaling ensures well-calibrated probabilities for rare-event detection, crucial in high-imbalance scenarios.

3. Hardware Feasibility:

- Quantify circuit-depth vs. noise trade-offs using simulation.
- If possible, partial demonstration on IBM Quantum or Rigetti to highlight NISQ-era viability (even if small-scale).

4. Foundation for T-KAN:

- Build scaffolding for future temporal (multi-time-slice) expansions, enabling LSTM-like gating within a KAN architecture.
- Document conceptual design for next-phase implementation.

Overall, this project aims to present a quantum-classical pipeline that leverages QKAN for LHC rare-event searches, with a strong emphasis on interpretability, calibration, and potential real-device applicability.

6 Resource Demands (Placeholder)

While most of the development can be done on local simulators and small HPC nodes, certain parts of the QKAN pipeline require dedicated resources:

- **Quantum Simulators:** For testing block-encoding, polynomial expansions, and quantum gradient updates, we will use PennyLane (CPU/GPU) and Qiskit Aer. Access to a stable Python 3.12 environment with at least 16GB of RAM.
- **NISQ Hardware (Optional):** If feasible, we will require cloud access to small quantum devices (IBM Quantum or Rigetti) to conduct partial circuit experiments. Such tests are queue-based and might involve limited shots/time.
- **GPU/CPU HPC Cluster:** For training on larger LHC samples, a multi-core CPU cluster or a GPU with sufficient VRAM (8–16GB) will speed up classical and hybrid optimization. If HPC resources are constrained, we will downscale data for proof-of-concept.
- **Software Stack:** Python 3.12, PennyLane, PyTorch/TensorFlow for classical layers, scikit-learn for calibration, plus Jupyter/VSCode for interactive development.

References

[1] P. Ivashkov *et al.*, *QKAN: Quantum Kolmogorov-Arnold Networks*, arXiv:2410.04435v1, (2024).

[2] E. E. Abasov *et al.*, *Application of Kolmogorov-Arnold Networks in high energy physics*, Moscow Univ. Phys. Bull. 79(8), (2024), [arXiv:2409.01724].

[3] E. Reinhardt, D. Ramakrishnan, and S. Gleyzer, *SineKAN: Kolmogorov-Arnold Networks using sinusoidal activation functions*, Front. Artif. Intell. 7:1462952, (2025).

[4] J. Erdmann, F. Mausolf, and J. L. Späh, *KAN we improve on HEP classification tasks? Kolmogorov-Arnold Networks applied to an LHC physics example*, arXiv:2408.02743, (2024).

[5] Z. Liu *et al.*, *KAN: Kolmogorov-Arnold Networks*, arXiv:2404.19756v5, (2025).

[6] R. Genet and H. Inzirillo, *Temporal Kolmogorov-Arnold Networks*, arXiv:2405.07344v3 (2024).