# Quantum-Assisted Anomaly Detection via Random Circuit Ensemble with Locality Sensitive Hashing

Jason Ludmir

12/13/2024

**Please note that since submitting the abstract, I had a major change in research direction for this project. The related code and text here is totally unrelated to what was discussed in my submitted abstract.**

## 1 Core Concept

Traditional anomaly detection algorithms face limitations with high-dimensional data, requiring extensive training phases or optimization procedures [1]. Quantum computing approaches to this problem currently demand either large qubit counts [3] or complex variational circuit training [2]. While Locality-Sensitive Hashing (LSH) has been combined with quantum computing, these efforts focus primarily on cryptographic applications (*e.g.*, quantum-resistant LSH schemes). The core challenge is development of a training-free quantum approach for dimensionality reduction with provable statistical guarantees. For example, classical autoencoders require extensive hyperparameter optimization (e.g. learning rate, network depth, activation functions etc.) and can overfit to normal data patterns.

I propose a hybrid quantum-classical technique using $k$ random-angle quantum circuits for feature reduction coupled with LSH-based anomaly detection. Each datapoint $x_i$ is processed through $k$ distinct circuits $C_j$ with randomly initialized parameters $\theta_j$, generating probability distributions $p_{i,j}$

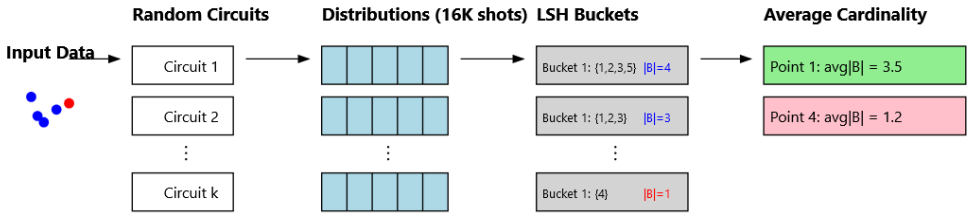



Figure 1: Quantum-Assisted Anomaly Detection Pipeline. The diagram illustrates the complete workflow from input data to anomaly detection. Input data points (blue: normal, red: anomalous) are processed through k different random quantum circuits. Each circuit generates a probability distribution from some number of measurements (in this example, 16K measurements, or shots as they are called in quantum computing). These distributions are then hashed into buckets using LSH, where $|B|$ denotes the bucket cardinality (number of elements). The median bucket cardinality ($median|B|$) across all circuits serves as the anomaly metric - normal points (like Point 1) appear in high-cardinality buckets, while anomalous points (like Point 4) typically appear in low-cardinality buckets (There is a typo in the image, it should state "Median Cardinality" not "Average Cardinality").

from $n$ measurements. These distributions are hashed via LSH, with median bucket cardinality across circuits serving as the anomaly metric (lower cardinality means fewer similar datapoints and thus a higher likelihood of anomalous behavior). Given the complexity of the technique and the amount of detail in the implementation, I cannot thoroughly explain the full process here in a couple of paragraphs. **I instead provide one appendix that gives a brief background on quantum computing (see Section 7) and another appendix (see Section 8) that explains the design of my technique in detail.**

## 2 Related Work

The recent work by Oh et al. (2024) [5] explores quantum implementations of LSH, but with a focus on cryptographic applications. Their work optimizes

quantum circuits for LSH hash functions to evaluate cryptographic algorithm resilience against quantum attacks. They achieve this through optimized quantum adders and parallelization techniques, resulting in improvements in circuit depth compared to previous approaches (circuit depth being a key metric for quantum circuits). Their main contribution is demonstrating how LSH can be efficiently implemented in quantum circuits, though their goal is cryptographic security analysis rather than anomaly detection.

Meira et al. (2022) [4] present LSHAD, a classical LSH-based approach to anomaly detection that is designed for large-scale datasets. Their method uses LSH to group similar data points and estimates density by analyzing bucket cardinalities - points in low-density buckets are flagged as anomalies. A key innovation is their automatic hyperparameter tuning mechanism that removes the need for manual parameter selection. While this work demonstrates LSH's effectiveness for anomaly detection, it operates purely in the classical domain without leveraging quantum advantages.

Neither of these works combines LSH-based anomaly detection with quantum computing in the way our approach proposes - using quantum circuits for feature reduction and dimensionality compression before classical LSH processing. Our method bridges this gap by leveraging quantum advantages in the data compression stage while maintaining the practical benefits of classical LSH for the final anomaly detection.

# 3 Hypothesis

**I hypothesize that using an ensemble of random quantum circuits for dimensionality reduction followed by LSH-based bucket cardinality analysis can provide effective anomaly detection, as quantum compression should preserve essential features for detecting outliers while LSH bucketing gives a natural, training-free way to identify anomalous points through their isolation in low-cardinality buckets.**

# 4 Experimental Setting

For evaluating our quantum-assisted anomaly detection approach, I will use three standard anomaly detection benchmark datasets: the Wisconsin Breast

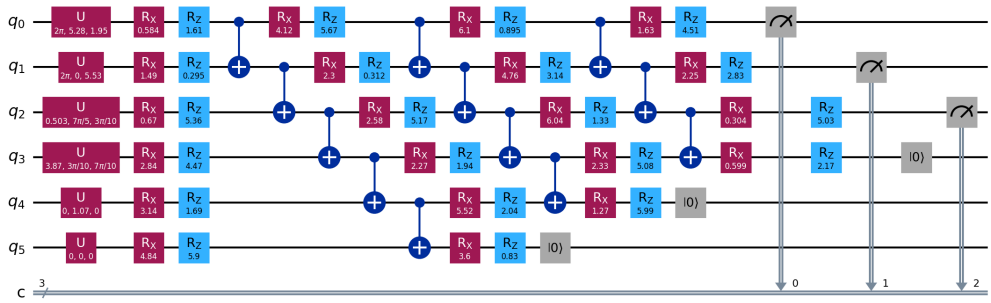



Figure 2: Quantum circuit showing the feature encoding and compression stages for the 16-feature Pen dataset. The circuit is read from left to right, with each horizontal lines each representing a qubit ($q_0$ - $q_5$). Purple U3 boxes show initial feature encoding gates with their three rotation angles, followed by alternating purple RX and blue RZ rotation gates with random parameters. Blue circles with vertical lines connecting qubits represent CNOT gates, where the filled circle is the control qubit and the $\oplus$ symbol indicates the target qubit. Gray boxes marked $|0\rangle$ represent qubit reset operations, while the rightmost meter symbols represent measurement operations. The bottom line marked 'c' with '3' represents a 3-bit classical register storing measurement outcomes.

Cancer dataset (367 samples, 30 features, with malignant samples being anomalous), the Pen-Based Recognition of Handwritten Digits dataset (809 samples, 16 features, with irregular pen strokes marked as anomalous), and the Letter Recognition dataset (1,600 samples, 32 features, with misformed letters labeled as anomalous). All of these can be found in the same widely-cited unsupervised anomaly detection paper [1]. For the Letter dataset, due to its large size, I took a random subset of 533 samples to use for experimentation. Each dataset comes with ground truth anomaly labels which are stripped during preprocessing to create an unsupervised learning scenario. Anomalies are detected by using a dynamic formula that takes the bottom $X\%$ of data samples by median bucket cardinality, with $X$ being some factor of the overall fraction of anomalies in the dataset.

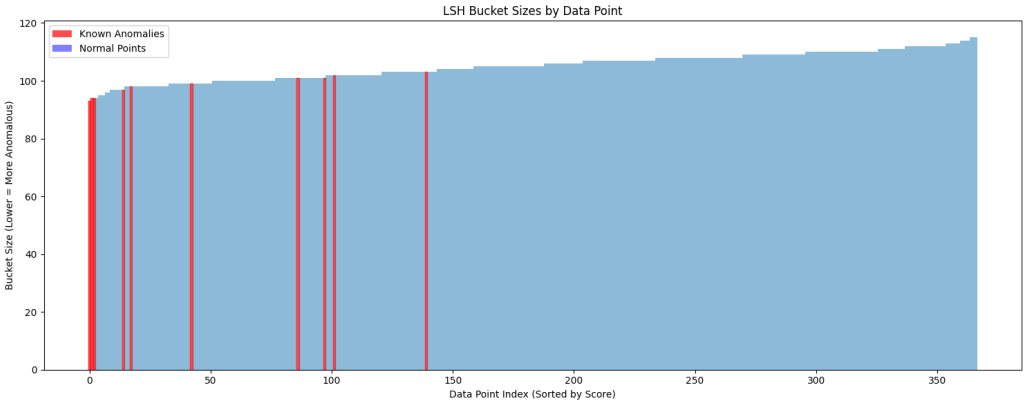
As our baseline, I compare against a classical unsupervised autoencoder. The autoencoder has an encoder pathway that compresses the input through

three dense layers (32, 16, and 8 neurons) with ReLU activation functions, followed by a symmetric decoder pathway that reconstructs the input through three dense layers (16, 32, and input dimension neurons), with the final layer using sigmoid activation. The model is trained using the Adam optimizer to minimize mean squared error loss between input and reconstruction. Training occurs over 100 epochs with a batch size of 32, using 10% of the training data for validation. The autoencoder is implemented using TensorFlow's Keras API and evaluates anomalies by comparing reconstruction errors against a threshold determined by the expected contamination rate in the dataset. This autoencoder will use reconstruction error as the anomaly score.

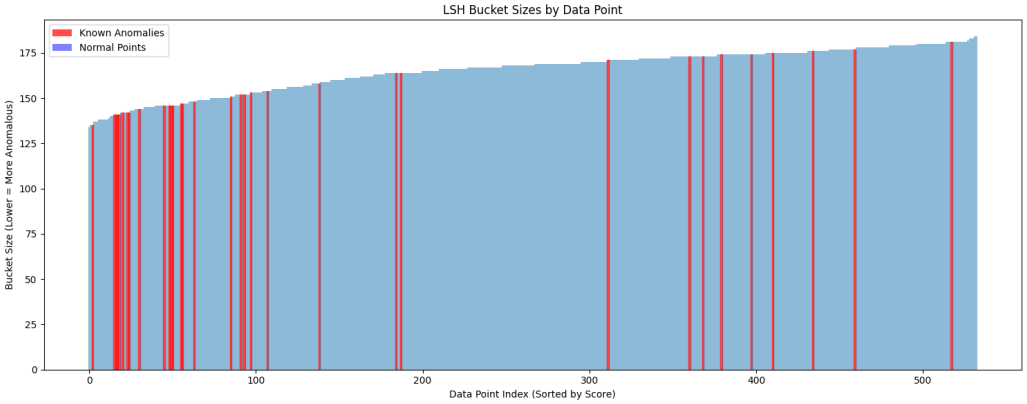
For evaluation metrics, I will use four complementary measures: **F1-score** to capture the overall detection performance balancing precision and recall, **precision** to measure the proportion of true anomalies among detected anomalies (reducing false positives), **recall** to measure the proportion of actual anomalies that were detected (reducing false negatives), and **accuracy** to measure overall correct label identification. I will use a simple mechanism to label anomalies: given an estimate percentage of total anomalies in the dataset $x\%$, we take the $x\%$ most anomalous (e.g. lowest median bucket sized-datapoints) and label them as anomalies; all other datapoints are labeled as normal. **For more details regarding implementation, please look at Section 8.**

# 5 Experimental Results

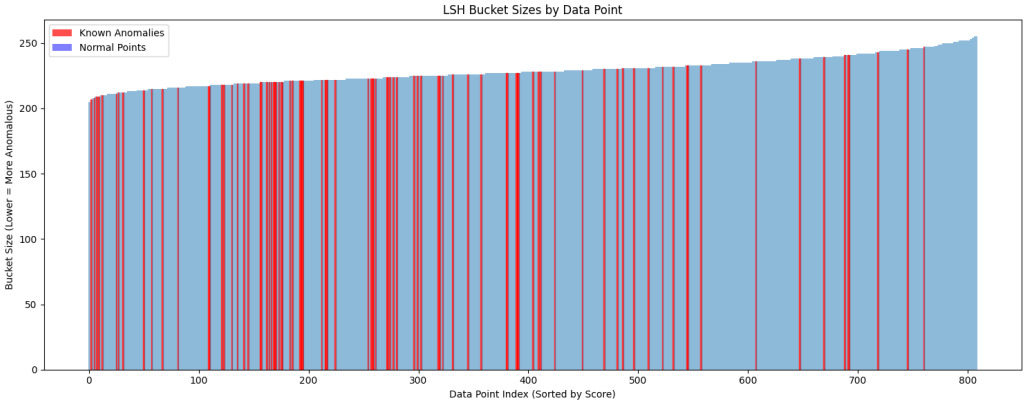
The results across the three datasets shown in Fig. 3 demonstrate the effectiveness of our quantum-LSH anomaly detection approach, with particularly strong performance on the Breast Cancer dataset. The plots show sorted median bucket sizes for each datapoint, where smaller median bucket sizes (left side) indicate more anomalous behavior. The Breast Cancer results are especially noteworthy, showing a clear concentration of known anomalies (red bars) on the left side of the plot where bucket sizes are smallest. This strong correlation between small bucket sizes and true anomalies suggests that our quantum measurement distributions effectively capture some of the distinctive patterns of anomalous data points, with these points consistently hashing to smaller buckets across iterations. While the Pen and Letter datasets show more dispersed anomaly distributions, they still maintain a visible trend toward smaller bucket sizes for anomalous points. **This clustering**




(a) Breast Cancer datapoints' anomaly scores.




(b) Letter datapoints' anomaly scores.




(c) Pen datapoints' anomaly scores.

Figure 3: Comparison of anomaly scores (median bucket sizes) across different datasets.

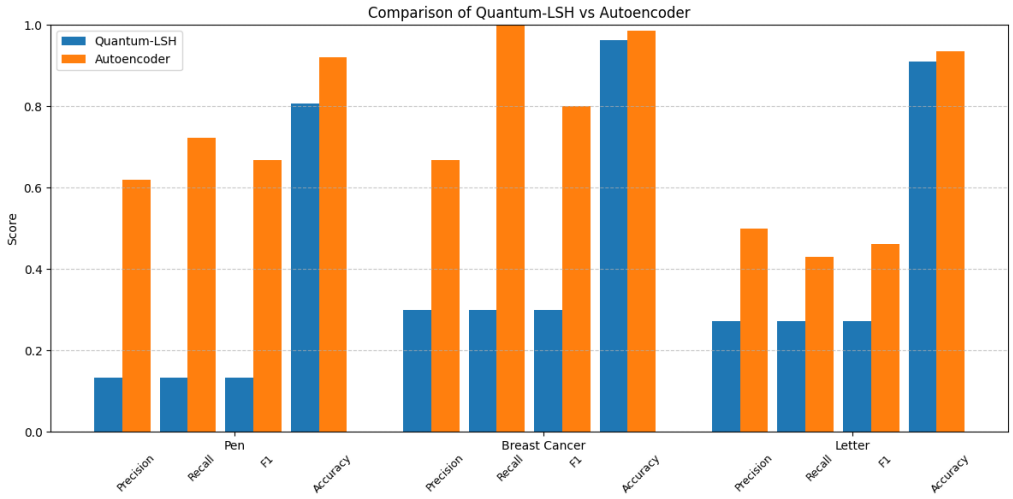



Figure 4: Final results comparing precision, recall, F1, and overall accuracy between our technique (Quantum-LSH) and a classical autoencoder.

**of anomalies in smaller buckets validates our core hypothesis that anomalous quantum measurement distributions would be less similar to other measurements, resulting in smaller bucket populations through the LSH process.**

The results in Fig. 4 show that our quantum-LSH approach unsurprisingly achieves lower anomaly detection performance compared to the classical autoencoder across all three datasets. However, it must be remembered that our quantum-LSH technique is not just performing anomaly detection, but also accomplishing significant dimensionality reduction through the quantum circuit compression from 7 qubits to 3 qubits, effectively reducing the feature space while maintaining reasonable detection capability. While it was expected that our novel approach would not outperform a well-established, highly-optimized technique like classical autoencoding, the comparable performance is promising, especially considering that our method shifts the most computationally intensive operations to quantum hardware. Moreover, there are a tremendous number of avenues of future exploration regarding this technique in terms of hyperparameters, circuit structures, LSH parameters, and more. The results demonstrate that our approach at least offers a viable quantum-classical hybrid solution for anomaly detection while providing the

additional benefit of built-in dimensionality reduction.

# 6 Justification of Hypothesis

The experimental results support both our initial problem framing and core hypothesis. On the problem side, we have demonstrated a viable quantum-classical hybrid approach that achieves dimensionality reduction and anomaly detection without requiring complex training procedures or large qubit counts. The LSH-based analysis of quantum measurement distributions provides a natural, training-free method for identifying anomalies, addressing the key limitations of existing approaches. Our performance metrics across three datasets, while not surpassing classical autoencoders, demonstrate comparable accuracy while offering built-in dimensionality reduction. More importantly, the clear clustering of anomalies in smaller LSH buckets strongly supports our hypothesis that quantum circuit compression preserves essential features for anomaly detection. The correlation between bucket size and anomaly status validates our theoretical framework that quantum measurement distributions of anomalous points would be measurably different from normal points, leading to their natural isolation in low-cardinality buckets through the LSH process.

# 7 Appendix A: Brief Background on Quantum Computing

A qubit is the fundamental unit of quantum computation, analogous to a classical bit but capable of existing in a superposition of states rather than just 0 or 1 (thus a qubit is in a probabilistic state during program execution with some probability of being in the 0 or 1 state, for example 60% in 0 and 40% in 1). Quantum gates are operations that manipulate these quantum states (i.e. alter these probabilities) through rotations and other transformations in the complex vector space of the qubit. Quantum entanglement is where the quantum states of multiple qubits become correlated in such a way that the state of each qubit cannot be described independently of the others - this is crucial as it allows information to be shared across all qubits in complex ways. For example, with two qubits we can represent four states: 00, 01, 10, 11, each with their own probabilities. Quantum cir-

cuits are sequences of quantum gates applied to one or more qubits, creating complex quantum states through a series of single-qubit gates (such as U3, RX and RZ) and multi-qubit entangling gates (such as the CNOT and CX two qubit gates). Due to the probabilistic nature of quantum computing explained above, measuring these quantum circuits yields probabilistic outcomes - when we measure a circuit with $n$ qubits, we obtain one of $2^n$ possible binary strings (e.g. For 3 qubits, '000', '001', etc.) according to probabilities determined by the quantum state at measurement time. To estimate these underlying probabilities, we must execute the same quantum circuit multiple times, with each execution called a "shot." In our implementation, each circuit is executed with 4096 shots, producing a dictionary of measurement counts (e.g., '000': 2006, '001': 1038, ...) that represents the sampled probability distribution of the quantum state.

# 8 Appendix B: Detailed Design Explanation

My system begins with a seven qubit quantum circuit that processes each data point through two distinct stages. The first stage implements angle encoding, where classical feature vectors are mapped to quantum states through single qubit $U_3$ gates, which provide three rotation angles ($\theta$, $\phi$, $\lambda$) per qubit. The implementation first normalizes all input features to the range $[0, 2\pi]$ using min-max normalization, specifically computing $\hat{x} = 2\pi\frac{x-x_{min}}{x_{max}-x_{min}}$ for each feature value $x$ ($2\pi$ is chosen as the scaling factor since it represents the maximum possible rotation angle for a qubit around any axis of the Bloch sphere). For each qubit $q_i$, the system takes up to three features from the normalized input vector $\hat{x}$ and assigns them to the three $U_3$ gate parameters. This approach differs from alternative quantum encoding schemes like amplitude embedding, which typically encodes $m$ features into $\log(n)$ qubits. While amplitude embedding can be more qubit-efficient, especially for high-dimensional data, it often requires deeper circuits with more gates to implement the necessary state preparations, making it more susceptible to decoherence errors on real quantum hardware. The angle embedding scheme was chosen specifically for its shallow circuit depth, which is particularly advantageous when running on local simulators (local simulators were used for my experimentation in this project). A future research direction would be to explore amplitude embedding implementations, which could yield better results by encoding more feature information into fewer qubits, though at

the cost of increased circuit complexity and runtime.

Following the embedding stage, the circuit implements a parameterized encoder section that transforms the encoded quantum state. Parameterized quantum gates are operations whose behavior is controlled by classical parameters - in our case, rotation angles $\theta_i$ that determine how much to rotate qubits around the X and Z axes of the Bloch sphere. The parameterized encoder structure begins with an initial layer of $R_x$-$R_z$ rotation pairs applied to every qubit, followed by compression layers that gradually reduce the active quantum state to a smaller number of qubits (in our case, our seven qubit circuit ends by reducing to three). Each compression layer consists of CNOT gates between adjacent qubits to create entanglement, followed by another set of $R_x$-$R_z$ rotation pairs on the remaining active qubits, and concludes by resetting the highest-indexed active qubit (see Fig.2). It should be noted here that this circuit structure is one among infintely many that could be tested; I chose this since there is a mix of parameters that are randomized as well as entanglement occuring between all qubits indirectly through the CNOT gates. However, another future research direction would be to attempt to use different encoder circuit structures.

Rather than optimizing the rotation parameters of the encoder through gradient descent or other variational methods, they are randomly initialized from a uniform distribution over $[0, 2\pi]$ for each ensemble member, creating different "views" of the encoded data. The choice to compress the quantum state to specifically three qubits is a design decision that balances computational efficiency with expressiveness - while more qubits would provide a higher-dimensional measurement space ($2^7$ possible output states for 7 qubits), three qubits yield an 8-dimensional probability vector ($2^3$). **This is one of the core benefits of our technique: We can perform dimensionality reduction using entanglement on our dataset and still retain anomaly detection capability.**

The implementation leverages parallel processing through a ThreadPoolExecutor to manage the computational workload efficiently. Since our method uses 1000 different random parameterizations of the encoder circuit to create diverse "views" of each datapoint, and since these iterations are completely independent of one another (making this an embarrassingly parallel problem), we can significantly reduce runtime by distributing these iterations across multiple threads. For each datapoint, the system creates and stores the angle embedding circuit once, then reuses it across all ensemble iterations - this optimization significantly reduces the computational overhead

by avoiding redundant circuit creation of the encoding stage, which remains constant across iterations. For each iteration thread, a new random parameterization of the encoder circuit (e.g. random angles for the parameterized gates) is generated and applied to all datapoints for the 4096 shots.

At this point, for each datapoint we have 1000 different quantum measurement distributions (each containing 4096 shots). Our hypothesis is that normal data points will produce similar distributions across these random circuit parameterizations, while anomalous points will produce consistently different patterns. We employ Locality-Sensitive Hashing to analyze the similarity structure, implemented in a Python class that uses an AND-OR amplification scheme with L=100 hash tables and K=6 hash functions per table.

Each quantum measurement distribution is first converted to an 8-dimensional normalized vector through our `counts_to_vector` function, which takes a dictionary of bitstring counts (e.g., {'000': 1024, '001': 512, ...}) and produces a consistent vector representation. Since we measure 3 qubits, we have $2^3 = 8$ possible bitstrings, and each component represents the relative frequency of observing that bitstring pattern.

Our LSH implementation maintains L=100 independent hash tables, with each table using K=6 randomly initialized hyperplanes stored in a (L, K, 8) dimensional array. When hashing a measurement vector v to a table t, we compute K dot products between v and the table's hyperplanes, converting the signs of these projections into a K-bit binary number that serves as the bucket index.

Mathematically, for a table t and measurement vector v, this computes: $h_t(v) = \sum_{i=1}^{K} 2^i \cdot [sign(r_{t,i} \cdot v) \geq 0]$ where $r_{t,i}$ is the i-th random hyperplane for table t.

This process repeats for all L tables, with each vector being assigned to exactly one bucket per table. The use of K hyperplanes per table (AND construction) makes the hashing more selective - vectors must agree on K different projections to hash to the same bucket. Having L independent tables (OR construction) then compensates for this strictness by providing multiple opportunities for similar vectors to collide.

The final anomaly score for each datapoint is computed as the median of all its observed bucket sizes across iterations. Lower scores suggest more anomalous behavior, as these points consistently hash to smaller buckets, indicating their measurement distributions are dissimilar from the majority of points. The use of median provides protection against outlier bucket sizes.

**This quantum-classical LSH framework provides an efficient way to identify points that consistently produce unusual quantum measurement distributions across the ensemble of random circuit parameterizations, without requiring explicit pairwise comparisons between all measurements and with a significant reduction in dimensionality.**

# 9 Appendix C: Code Structure and Execution

The implementation of the quantum processing component of the project consists of four main Python modules: `main.py` for orchestrating the experimental pipeline, `angle_encoding.py` in the "Embedding" directory for feature encoding into quantum states, `rx_rz_ansatz.py` in the "Ansatzes" for quantum circuit construction, and the preprocessing module for the datasets (e.g., in "Preprocessing", `goldstein_uchida_preprocess.py`). The codebase requires Qiskit and standard scientific Python libraries (NumPy, Pandas). The "Data" directory contains the three datasets as CSV files.

To execute experiments, run `python main.py NUM_QUBITS [--num_threads N]`, where `NUM_QUBITS` should be set to $\lceil \text{num\_features}/3 \rceil$. The program creates an ensemble of random quantum circuits, processes the dataset through these circuits using parallel execution, and saves results to `results/ensemble_res.pkl`.
To execute the experiments the same way I did to generate the results in this project, run: `python main.py 7 --num_threads 8`
Make sure to change the filepath variable in main.py to use the dataset desired.

For the LSH portion of the code, after the above quantum processing component is complete, run `python lsh_extract.py`. This will take the result saved to `results/ensemble_res.pkl` and run the LSH algorithm. Once this is complete, there will be a printed message stating the results of the LSH run (such as summary statistics of the behavior of anomalies and normal datapoints with regard to bucket sizes), and a file called "similarity_scores.png" will be created to show the results for that dataset.

`python classical_autoencoder.py` executes an independent module that runs the comparable autoencoder; as with before, make sure to change the filepath to whatever datafile you desire before running. It will print summary

results for the autoencoder's runs.

Finally, there is `FinalPlotter.py`. It creates the combined plot comparing the results from running the two comparable methods. It uses the outputs printed from the classical autoencoder runs and the LSH runs (manually copied into this file due to the small number of values).

***Acknowledgment: Claude LLM was used to help revise both this document and related code.**

# References

[1] Markus Goldstein and Seiichi Uchida. A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PLOS ONE*, 11(4):1–31, 04 2016.

[2] Moe Hdaib, Sutharshan Rajasegarar, and Lei Pan. Quantum deep learning-based anomaly detection for enhanced network security. *Quantum Machine Intelligence*, 6(1):26, May 2024.

[3] Alon Kukliansky, Marko Orescanin, Chad Bollmann, and Theodore Huffmire. Network anomaly detection using quantum neural networks on noisy quantum computers. *IEEE Transactions on Quantum Engineering*, 2024.

[4] Jorge Meira, Carlos Eiras-Franco, Verónica Bolón-Canedo, Goreti Marreiros, and Amparo Alonso-Betanzos. Fast anomaly detection with locality-sensitive hashing and hyperparameter autotuning. *Information Sciences*, 607:1245–1264, 2022.

[5] Yujin Oh, Kyungbae Jang, and Hwajeong Seo. Quantum implementation of LSH. Cryptology ePrint Archive, Paper 2024/1082, 2024.