Generative Modeling of VQ-VAE Latent Spaces from LBM Fluid Simulations: A Quantum-Classical Comparison

Achraf Hsain
Al Akhawayn University
a.hsain@aui.ma

Fouad Mohammed Abbou
Al Akhawayn University
f.abbou@aui.ma

Lamiae Bouanane
Al Akhawayn University
l.bouanane@aui.ma

Mouna Kettani
Al Akhawayn University
m.kettani.ma

***Abstract*—This paper addresses the challenge of efficiently modeling and generating complex fluid dynamics data by exploring the application of classical and quantum generative models to a compressed latent space representation. We developed a GPU-accelerated Lattice Boltzmann Method (LBM) simulator to generate high-fidelity fluid vorticity data, which was subsequently encoded into a discrete, low-dimensional latent space using a Vector Quantized Variational Autoencoder (VQ-VAE). The core of this research involves a comparative analysis of a classical Long Short-Term Memory (LSTM) network against two quantum generative models, the Quantum Circuit Born Machine (QCBM) and the Quantum Generative Adversarial Network (QGAN), for learning and sampling from this discrete latent space distribution. Our results demonstrate that the quantum generative models consistently outperformed the classical LSTM baseline in accurately capturing the latent space distribution, as evidenced by quantitative metrics including average minimum distance and nearest neighbor analysis, complemented by qualitative visualizations (t-SNE, PCA). Notably, the QCBM exhibited superior performance among the models evaluated. These findings highlight the potential of current quantum computing approaches, even when simulated, to offer tangible advantages over established classical techniques for complex data generative tasks in scientific domains. This work contributes empirical evidence supporting the integration of quantum machine learning with classical dimensionality reduction for enhanced analysis and synthesis of intricate scientific data, paving the way for more efficient computational fluid dynamics and accelerated scientific discovery.**

***Index Terms*—Computational Fluid Dynamics, Generative Models, Quantum Machine Learning, Lattice Boltzmann Method, Variational Autoencoder, Quantum Circuit Born Machine, Quantum Generative Adversarial Network.**

I. Introduction

Computational Fluid Dynamics (CFD) plays a crucial role in understanding and predicting the behavior of fluid systems across a wide range of scientific and engineering disciplines, including aerospace, automotive, environmental science, and energy [1], [2]. Traditional methods for solving fluid dynamics problems, such as those based on the Navier-Stokes equations [21]–[23], often involve significant computational resources, particularly for complex or turbulent flows [3]. The increasing complexity of simulations and the resulting high-dimensional data necessitate the development of more efficient methods for data representation, analysis, and generation. Generative models have emerged as powerful tools for learning the underlying probability distributions of complex datasets and generating novel, synthetic data samples that resemble the original data [4]–[6]. These models can capture intricate patterns and structures within data, offering potential benefits for tasks such as data augmentation, anomaly detection, and understanding complex system dynamics. Dimensionality reduction techniques are often employed to handle the high-dimensional output of simulations. Autoencoders and Variational Autoencoders (VAEs) are neural network architectures capable of learning compressed, lower-dimensional latent space representations of data [6]–[8]. VAEs, in particular, provide a probabilistic framework for encoding data into a continuous latent space, enabling the generation of new data by sampling from this learned distribution [6], [10]. Extending this concept, Vector Quantized Variational Autoencoders (VQ-VAEs) learn a discrete latent space represented by a codebook of embeddings, which can be particularly advantageous for tasks requiring discrete inputs or for integration with certain computational paradigms [12]. Concurrently, the field of quantum computing is rapidly advancing, demonstrating potential advantages for specific computational problems, including certain machine learning tasks [13]. Quantum machine learning algorithms, such as Quantum Circuit Born Machines (QCBMs) and Quantum Generative Adversarial Networks (QGANs), leverage the principles of quantum mechanics to potentially model complex probability distributions more effectively than classical methods [13]–[16]. Exploring the intersection of quantum computing and CFD data analysis, particularly through generative modeling of compressed latent spaces, presents an exciting avenue for research. This paper investigates the application of classical and quantum generative models to the latent space representation of fluid vorticity data. Specifically, we address the challenge of efficiently modeling and sampling from the complex distribution of fluid flow patterns captured in a low-dimensional, discrete latent space. We propose an approach that combines a GPU-accelerated Lattice Boltzmann Method (LBM) simulator for data generation, a VQ-VAE for learning a discrete latent representation of fluid vorticity, and a comparative study of classical Long Short-Term Memory (LSTM) networks, QCBMs, and QGANs for modeling and sampling from this latent space prior distribution. The key contributions of this work include:

- The application of a Vector Quantized Variational Autoencoder (VQ-VAE) to create a compressed, discrete latent space representation of 2D fluid vorticity profiles.
- A comparative analysis of classical (LSTM) and quantum (QCBM ,QGAN) generative models for learning and sampling from the VQ-VAE's latent space distribution.
- Providing insights into the potential of current quantum computing approaches, even when simulated, for capturing complex structures in fluid dynamics data compared to established classical machine learning methods.

The remainder of this paper is organized as follows. Section II provides a detailed background on the theoretical concepts underpinning this research, including fluid dynamics, the Lattice Boltzmann Method, Variational Autoencoders, Vector Quantized VAEs, and relevant quantum computing principles. Section IV describes the methodology employed, covering the LBM simulator implementation, VQ-VAE architecture and training, and the design and training of the classical and quantum generative models. Section V presents the experimental results and analysis, including visualizations and quantitative comparisons of the generative models' performance. Section VI discusses the implications of our findings. Finally, Section VIII concludes the paper.

II. Literature Review

This section provides a comprehensive overview of the foundational concepts and prior research relevant to this work, spanning computational fluid dynamics, generative modeling, and quantum machine learning.

A. Fluid Dynamics and Computational Fluid Dynamics (CFD)

Fluid mechanics is a fundamental branch of physics concerned with the motion of fluids (liquids, gases, and plasmas) and the forces acting upon them. The behavior of Newtonian fluids is primarily governed by the Navier-Stokes equations, a set of partial differential equations describing the conservation of momentum and mass [21]–[23]. These equations are notoriously difficult to solve analytically, especially for complex geometries or turbulent flow regimes. Turbulence, characterized by chaotic, unpredictable fluid motion, remains one of the most challenging problems in classical physics and engineering [3]. Computational Fluid Dynamics (CFD) emerged as a discipline to tackle these challenges by employing numerical methods and algorithms to simulate fluid flows [1], [2]. CFD has become an indispensable tool in diverse fields, enabling the design and analysis of aircraft wings, optimization of automotive aerodynamics, prediction of weather patterns, and simulation of blood flow in biological systems. Common numerical approaches in CFD include Finite Difference Methods (FDM), Finite Volume Methods (FVM), and Finite Element Methods (FEM). The choice of method often depends on the specific problem characteristics, computational resources, and desired accuracy. A key dimensionless parameter in fluid dynamics is the Reynolds number (Re), which represents the ratio of inertial forces to viscous forces within a fluid [23]. It is a critical indicator of flow regime; low Reynolds numbers typically correspond to laminar (smooth, orderly) flow, while high Reynolds numbers are associated with turbulent (chaotic) flow. Accurate simulation of high Reynolds number flows remains computationally intensive, driving the need for more efficient simulation techniques and data analysis methods.

B. Lattice Boltzmann Method (LBM)

The Lattice Boltzmann Method (LBM) is an alternative CFD approach that has gained significant popularity due to its conceptual simplicity, computational efficiency, and suitability for parallel implementation [17]. Unlike traditional CFD methods that directly discretize macroscopic conservation equations like Navier-Stokes, LBM is based on a simplified kinetic model. It simulates the statistical behavior of a large number of fluid particles on a discrete lattice grid, tracking the evolution of particle distribution functions. Macroscopic properties such as density, velocity, and pressure are then derived from moments of these distribution functions. The LBM typically involves two main steps: collision and streaming. In the collision step, particle distribution functions at each lattice node relax towards a local equilibrium distribution, often modeled by the Bhatnagar-Gross-Krook (BGK) approximation. The streaming step involves the movement of particles to neighboring lattice nodes according to their discrete velocities. The lattice structure is defined by the DnQm notation, where 'n' is the number of spatial dimensions and 'm' is the number of discrete velocity vectors connecting a node to its neighbors. Commonly used schemes include D2Q9 for 2D simulations (2 dimensions, 9 velocities) and D3Q19 or D3Q27 for 3D simulations [17]. The D2Q9 model, used as the primary structure for 2D simulations in this work, is well-suited for simulating incompressible or nearly incompressible flows. LBM's inherent locality and explicit time-stepping make it highly amenable to parallel processing, particularly on Graphics Processing Units (GPUs), enabling significant acceleration compared to CPU-based implementations for large-scale simulations [18]. Boundary conditions, such as no-slip boundaries (e.g., bounce-back method) and inflow/outflow conditions, are implemented to define the interaction of the fluid with solid objects and domain boundaries [19], [20].

C. Autoencoders and Variational Autoencoders (VAEs)

Dimensionality reduction is a critical step in analyzing and processing high-dimensional data, including the outputs of CFD simulations. Autoencoders are a class of artificial neural networks designed to learn efficient data encodings [7], [8]. An autoencoder consists of an encoder network that maps the input data to a lower-dimensional latent space representation and a decoder network that reconstructs the original data from this latent representation. The network is trained to minimize the reconstruction error. Variational Autoencoders (VAEs) extend the autoencoder framework by introducing a probabilistic perspective [6]. Instead of mapping an input to a fixed point in the latent space, the encoder in a VAE maps it to parameters (typically mean and variance) of a probability

distribution (often assumed to be Gaussian) in the latent space. This probabilistic encoding allows VAEs to generate new data samples by sampling from the learned latent distributions and passing these samples through the decoder. VAEs are trained by optimizing a lower bound on the data likelihood, known as the Evidence Lower Bound (ELBO), which incorporates both the reconstruction loss and a regularization term (Kullback-Leibler divergence) that encourages the latent distributions to be close to a prior distribution (e.g., a standard Gaussian) [6], [10], [25].

D. Vector Quantized Variational Autoencoders (VQ-VAEs)

While VAEs learn a continuous latent space, some applications benefit from a discrete latent representation. The Vector Quantized Variational Autoencoder (VQ-VAE) modifies the standard VAE by replacing the continuous latent space with a discrete codebook of learned embedding vectors [12]. The encoder in a VQ-VAE outputs a representation that is then "quantized" by finding the closest embedding vector in the codebook. This discrete codebook vector is then passed to the decoder. The training of a VQ-VAE involves minimizing a loss function that typically includes three components: a reconstruction loss (measuring the quality of the decoded output), a codebook loss (updating the embeddings in the codebook based on the encoder outputs), and a commitment loss (encouraging the encoder output to stay close to the chosen codebook vector). A technique called "straight-through estimation" is often used during backpropagation to handle the non-differentiable quantization step. VQ-VAEs have been successfully applied in various domains, including image generation, video prediction, and speech synthesis, due to their ability to learn rich discrete representations. Their capacity to produce discrete latent codes makes them particularly relevant for integration with models that require discrete inputs, such as certain quantum algorithms [12].

E. Quantum Computing Fundamentals

Quantum computing is a paradigm that leverages the principles of quantum mechanics to perform computations. Unlike classical bits that represent either 0 or 1, quantum bits, or qubits, can exist in a superposition of both states simultaneously. The state of a system of n qubits exists in a 2^n-dimensional complex vector space, allowing for the representation of exponentially large state spaces. Quantum gates, analogous to classical logic gates, are unitary operations that manipulate the states of qubits. Sequences of quantum gates form quantum circuits, which perform the desired computation. A key quantum phenomenon is entanglement, where the states of two or more qubits become correlated in a way that cannot be described classically. Entanglement is a crucial resource for many quantum algorithms and enables quantum computers to potentially outperform classical computers on certain tasks. Measurement is the process of obtaining a classical outcome from a qubit's state, which collapses the superposition according to probabilities determined by the quantum state. While large-scale fault-tolerant quantum computers are still under development, noisy intermediate-scale quantum (NISQ) devices are currently available [13]. Furthermore, classical simulators can emulate the behavior of quantum circuits for a limited number of qubits, providing a platform for algorithm development and testing.

F. Quantum Generative Models

Leveraging the ability of quantum systems to naturally represent and manipulate high-dimensional probability distributions, research has explored the development of quantum generative models. These models aim to learn a target probability distribution and generate samples from it using quantum circuits. The Quantum Circuit Born Machine (QCBM) is a type of quantum generative model where a parameterized quantum circuit is used to define a probability distribution over computational basis states [14]. The parameters of the circuit are trained to minimize a distance measure between the distribution sampled from the quantum circuit and the target data distribution. Training QCBMs often involves optimizing these parameters using classical optimization techniques, guided by a loss function computed from measurements on the quantum circuit [14]. The Maximum Mean Discrepancy (MMD) is a commonly used loss function for training generative models, including QCBMs, as it provides a kernel-based measure of the distance between two distributions [26]. The squared MMD between distributions p and q with kernel k is given by:

$$\text{MMD}^2(p, q) = \mathbb{E}_p[k(x, x')] - 2\mathbb{E}_{p,q}[k(x, y)] + \mathbb{E}_q[k(y, y')]$$

where $\mathbb{E}$ denotes the expected value. The Radial Basis Function (RBF) kernel is a popular choice for k [26]. Quantum Generative Adversarial Networks (QGANs) adapt the classical GAN framework to the quantum domain [15], [16]. A QGAN typically consists of a quantum generator network that produces quantum states (or classical samples derived from them) and a discriminator (which can be classical or quantum) that tries to distinguish between real data samples and samples from the generator. The generator and discriminator are trained in an adversarial manner, where the generator aims to fool the discriminator, and the discriminator aims to correctly classify samples [5], [15], [16]. Implementing QGANs on current hardware or simulators faces challenges related to qubit connectivity, circuit depth, and the number of qubits required, especially for high-dimensional data. Techniques like patching or splitting the problem across multiple smaller quantum circuits can be employed to mitigate these limitations [13].

G. Relevant Machine Learning Techniques

Classical machine learning models also play a significant role in generative modeling and sequence prediction tasks. Long Short-Term Memory (LSTM) networks are a type of recurrent neural network (RNN) particularly well-suited for modeling sequential data [24]. LSTMs utilize gating mechanisms to mitigate the vanishing gradient problem, allowing them to learn long-range dependencies in sequences. They have been widely applied in natural language processing,

time series prediction, and sequence generation. When training autoregressive models like LSTMs for sequence generation, a technique called "teacher forcing" is often used [28], [29]. Teacher forcing involves feeding the true previous output from the training dataset as the input to the model at the current time step, rather than using the model's own prediction from the previous step. This can stabilize and accelerate training, particularly in the early stages, by preventing the accumulation of errors. The comparative analysis in this work utilizes techniques for visualizing and comparing high-dimensional data distributions. T-distributed Stochastic Neighbor Embedding (t-SNE) is a non-linear dimensionality reduction technique commonly used for visualizing high-dimensional data by giving each data point a location in a two or three-dimensional map [27]. Principal Component Analysis (PCA) is a linear dimensionality reduction technique that transforms data to a new coordinate system such that the greatest variance by some projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate, and so on [8]. These visualization methods are valuable for gaining intuition about the structure and relationships within complex datasets and the distributions learned by generative models. Metrics such as average minimum distance and nearest neighbor analysis provide quantitative measures for comparing how well generated samples match the distribution of a reference dataset [25], [26].

III. Problem Statement and Objectives

As highlighted in the literature review, traditional Computational Fluid Dynamics (CFD) simulations, particularly for complex phenomena like turbulent flows at high Reynolds numbers, are computationally intensive and generate vast amounts of high-dimensional data [3]. Analyzing, storing, and leveraging this data effectively poses significant challenges. While dimensionality reduction techniques and classical generative models offer potential solutions for data compression and synthesis, their application to the intricate patterns of fluid dynamics, specifically in a compressed latent space, warrants further investigation. Furthermore, the nascent field of quantum computing presents new possibilities for tackling computational problems that are intractable for classical computers. Quantum machine learning algorithms, particularly generative models, hold promise for modeling complex probability distributions that may be difficult for classical counterparts [13]. However, the practical utility and performance of current quantum generative models, especially when applied to real-world or simulation-derived scientific data like fluid dynamics, remain active areas of research. It is unclear whether existing quantum approaches, even when simulated on classical hardware, can offer a discernible advantage over established classical machine learning techniques for tasks such as learning and sampling from the latent space of fluid flow data. The core problem addressed in this research is the need for efficient and effective methods to model and generate representative data from the complex, high-dimensional outputs of fluid dynamics simulations. Specifically, we aim to explore whether quantum generative models can provide superior performance compared to classical methods in learning and sampling from a compressed, discrete latent space representation of fluid vorticity. Based on this problem statement, the primary objectives of this study are:

- To develop a robust and efficient computational framework capable of generating physically accurate fluid dynamics data, specifically focusing on fluid vorticity profiles using a GPU-accelerated Lattice Boltzmann Method (LBM) simulator.
- To apply a Vector Quantized Variational Autoencoder (VQ-VAE) to learn a highly compressed and discrete latent space representation of the generated fluid vorticity data, thereby reducing the dimensionality while preserving essential flow characteristics.
- To train and evaluate classical and quantum generative models on the learned discrete latent space to model its underlying prior distribution.
- To conduct a comprehensive comparative analysis of the performance of the trained classical and quantum generative models in terms of their ability to accurately capture and sample from the latent space distribution, using both visualization techniques [27] and quantitative metrics [25], [26].
- To assess the potential advantages and limitations of using current quantum computing approaches (simulated) for generative modeling tasks in the context of fluid dynamics data compared to established classical methods.

By achieving these objectives, this research seeks to contribute to the understanding of how advanced machine learning techniques, including emerging quantum algorithms, can be leveraged to address the challenges of analyzing and generating complex scientific data, potentially paving the way for more efficient simulation and data-driven discovery in fluid dynamics and other fields.

IV. Methodology

This section details the computational framework and experimental procedures employed in this research, encompassing the generation of fluid dynamics data, the learning of a compressed latent space representation, and the subsequent training and evaluation of generative models. The methodology follows a modular and iterative approach, allowing for independent development and testing of each core component. All computational experiments were performed using an NVIDIA RTX 4060 GPU. The software was implemented primarily in Python, utilizing libraries including PyTorch, PennyLane, Scikit-learn, NumPy, Open3D, Matplotlib, and PyVista.

A. LBM Fluid Simulator and Data Generation

Fluid dynamics data for this study was generated using a custom GPU-accelerated Lattice Boltzmann Method (LBM) simulator, implemented in Python to leverage the computational power of GPUs for enhanced performance [18]. The

simulator was capable of both 2D and 3D simulations, although the primary dataset for this research was generated using the 2D implementation.

The simulation framework managed the execution flow and configuration of parameters such as grid dimensions, fluid properties, and object geometry. Object geometry was incorporated into the simulation grid via a masking process. For 2D simulations, object shapes were defined from image data to create solid boundaries within the grid. For 3D simulations, 3D models were converted into a voxel-based representation. Visualization capabilities were integrated to render flow fields such as velocity, vorticity, and density.

The core LBM simulation logic implemented the fundamental steps, typically residing on the GPU for accelerated computation:

- **Inflow**: Sets the velocity boundary condition at the designated inlet region of the simulation grid, updating particle distribution functions to reflect the incoming fluid velocity profile [19].
- **Compute Equilibrium**: Calculates the local equilibrium distribution function at each grid node based on macroscopic fluid properties, appropriate for the chosen LBM scheme (e.g., D2Q9) [17].
- **Collide**: Relaxes particle distribution functions towards the local equilibrium, modeling inter-particle collisions governed by the relaxation parameter (τ) related to fluid viscosity [17].
- **Bounce Back**: Implements the no-slip boundary condition at solid boundaries, enforcing zero fluid velocity at these surfaces. Circular boundary conditions were also used for domain periodicity [20].
- **Propagate**: Performs the streaming step, moving particle distribution functions to adjacent nodes according to their discrete velocities, conserving mass and momentum [17].

The simulator's physical accuracy was benchmarked against established CFD results for standard test cases [17]. Data collection involved systematically saving snapshots of the fluid vorticity grid at specified time steps after an initial warm-up period. The dataset used for training the VQ-VAE consisted of 2D vorticity profiles from a simulation of flow around a cylinder with a radius of 16 pixels, an inlet flow velocity of 0.1 Mach, and a Reynolds number of 500. Each snapshot represented the vorticity field as a grid of size 256×64. A total of 1999 simulation snapshots were collected for the dataset.

B. VQ-VAE Latent Space Learning

To obtain a compressed, discrete latent space representation of the fluid vorticity data, a Vector Quantized Variational Autoencoder (VQ-VAE) model was employed [12]. The VQ-VAE model architecture, defined in a dedicated Python module, consists of three main components:

- **Image Encoder**: This module processes the 256×64 input vorticity grid. It comprises a series of 2D convolutional layers, each followed by a Rectified Linear Unit (ReLU) activation function and batch normalization.

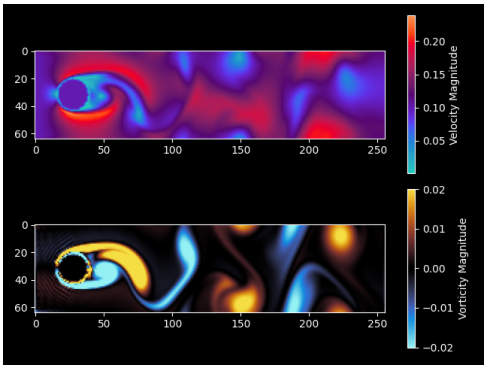

Fig. 1. Snapshot From The LBM Generated Dataset

The convolutional layers utilize a kernel size of 3×3, a stride of 2, and padding of 1, progressively reducing the spatial dimensions while increasing the number of feature maps. This process continues until a feature space of size $256 \times 16 \times 4$ is obtained (256 being the number of channels).

- **Quantized Encoder-Decoder**: This module bridges the encoder and decoder. It first uses Multi-Layer Perceptrons (MLPs) to map the $256 \times 16 \times 4$ feature space into a 2048-dimensional vector. This vector is then further reduced by another MLP to a 7-dimensional vector, representing the model's continuous latent space output. This continuous vector is then quantized by finding the closest embedding vector in a discrete codebook containing 128 codewords, each with a dimension of 7. A separate MLP maps the selected 7-dimensional codeword back to a 2048-dimensional vector, which serves as the input to the decoder. ReLU activation and batch normalization are applied within this module.
- **Image Decoder**: This module mirrors the encoder, reconstructing the original vorticity grid from the quantized latent representation. It consists of a series of transposed convolutional layers with corresponding kernel sizes, strides, and padding to invert the encoding process and output a 256×64 grid.

$$\mathcal{L}_{\text{VQ-VAE}} = \underbrace{\|x - \hat{x}\|_2^2}_{\text{Reconstruction Loss}} + \underbrace{\|\text{sg}[z_e(x)] - e\|_2^2}_{\text{Codebook Loss}} + \beta \underbrace{\|z_e(x) - \text{sg}[e]\|_2^2}_{\text{Commitment Loss}}$$

The VQ-VAE model was trained using the Adam optimizer with a learning rate of 0.0005 for 50 epochs, with a batch size of 32. The training objective minimized a custom loss function comprising three components: a normalized Mean Squared Error (MSE) reconstruction loss, a codebook loss, and a commitment loss weighted by a factor $\beta = 0.2$. Straight-through estimation was utilized to enable gradient flow through the non-differentiable quantization step during training. The learned discrete latent space, represented by the VQ-VAE's codebook, serves as the target distribution for the subsequent generative modeling tasks [12].

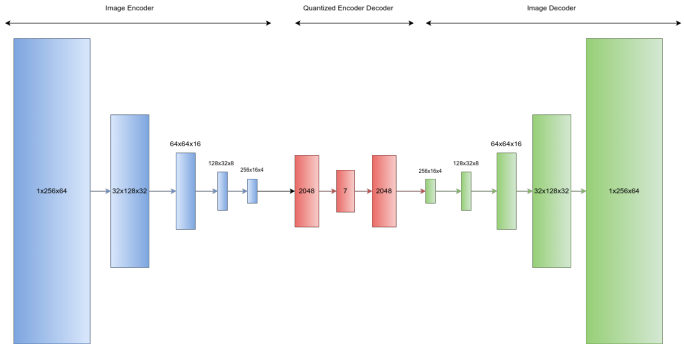

Fig. 2. VQ-VAE Architecture

C. Generative Modeling of Latent Space

Following the learning of the discrete latent space using the VQ-VAE [12], classical and quantum generative models were employed to model the prior distribution of this latent space. The objective was to learn a model capable of generating novel latent vectors that are representative of the distribution observed in the VQ-VAE codebook. Three different models were implemented and evaluated: a Quantum Circuit Born Machine (QCBM) [14], a Quantum Generative Adversarial Network (QGAN) [15], [16], and a classical Long Short-Term Memory (LSTM) network [24] as a baseline.

1) Quantum Circuit Born Machine (QCBM): The Quantum Circuit Born Machine (QCBM) implementation leveraged the PennyLane Python library, which facilitates differentiable programming for quantum computing and supports GPU acceleration for simulations [13], [14]. The approach involved training seven independent QCBM models, each dedicated to modeling the empirical distribution of a single dimension (x_i) of the 7-dimensional latent space vector, where $i \in \{1, \ldots, 7\}$. Quantum models typically require discrete inputs. To accommodate this, the continuous float values of the latent space dimensions were discretized using a custom Gaussian quantization function. Based on the observed approximately

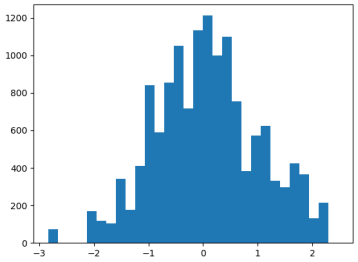

Fig. 3. Continious Latent Space Distribution

Gaussian distribution of the latent space values, this function mapped the values into 256 bins according to a Gaussian curve. Each bin value was then encoded into an 8-bit binary string, serving as the target output quantum state for the QCBMs. Each of the seven independent QCBM circuits was designed to take 8 qubits initialized in the $|0\rangle$ state. The circuit architecture consisted of 7 layers. Each layer comprised single-qubit Ry rotation gates applied to all 8 qubits, followed by a pattern of circular entanglement using Controlled-Z (CZ) gates [13], [14]. The output of the quantum circuit, obtained through measurements of the 8 qubits, yields a probability distribution over the 256 possible computational basis states, intended to approximate the empirical distribution of the corresponding latent dimension. Training of the QCBM models was performed using PennyLane's PyTorch interface and the lightning.gpu device for GPU-accelerated simulation [13]. The circuit parameters were optimized using the Adam optimizer with a learning rate of 0.1 over 100 iterations. The parameter-shift rule was employed for computing gradients with respect to the circuit parameters [14]. The loss function minimized was the squared Maximum Mean Discrepancy (MMD) between the probability distribution sampled from the QCBM and the target empirical distribution of the latent dimension [26]. A Radial Basis Function (RBF) kernel was used for the MMD calculation, with Gaussian bandwidths of 0.25, 0.5, and 1.0, which are common choices in quantum machine learning literature [26].

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right)$$

The trained model weights for each of the seven QCBMs were saved independently. For sampling, a `QCBMSampler` class was developed to load the trained weights and generate a full 7-dimensional latent vector by sampling independently from each of the seven trained QCBMs.

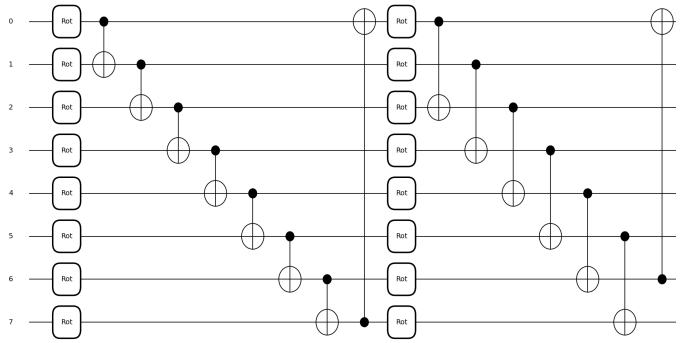

Fig. 4. 2 Layers From QCBM Architecture

2) Quantum Generative Adversarial Network: A QGAN was also implemented to model the latent space distribution, adapting the adversarial training paradigm [5], [15], [16]. The QGAN architecture consisted of a classical discriminator and a quantum generator. The Discriminator was a classical feed-forward neural network comprising three layers. It received a flattened input representing the probability distribution across the 256 bins for each of the 7 latent dimensions (256×7 matrix). This input was mapped to a single scalar output, followed by a sigmoid activation function, indicating the probability that the input distribution was real (from the VQ-VAE codebook) rather than generated. The Generator component

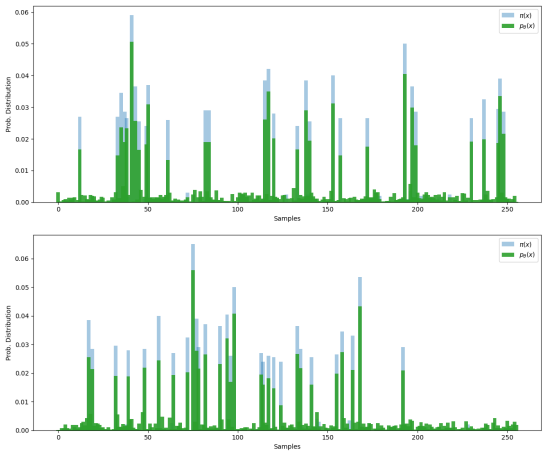

Fig. 5. QCBM Learned Distributions for layer 1 and 7

consisted of a collection of quantum circuits—specifically, seven independent circuits. All seven outputs are combined into a single codeword before applying backpropagation across the entire system. Each generator used 10 qubits: 8 for the actual input (discretized latent values) and 2 ancillary qubits initialized to $|0\rangle$, which the model can utilize for entanglement and interaction with the input. Each circuit consisted of 6 layers, with each layer composed of Pauli-Ry rotation gates for encoding and trainable parameters, followed by full circular entanglement using Controlled-Z gates [13], [15], [16]. The outputs from the seven independent quantum generators were combined to form a complete 7-dimensional latent vector. The discriminator and each generator were trained using separate Adam optimizers with a learning rate of 0.01, a batch size of 32, and trained for 2 epochs. The training followed an adversarial process. The discriminator was trained to minimize a binary cross-entropy loss, distinguishing between real and generated latent vectors. The generator was trained to maximize the discriminator's output for generated samples (a modified loss function maximizing $\log D(x)$ instead of minimizing $1-\log D(x)$ for improved stability) [5], [15], [16].

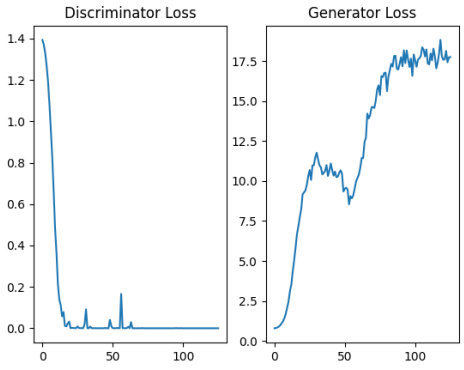

Fig. 6. QGAN Training Losses

At inference time, the model takes in random noise as input, which is bin-discretized before being passed through the generator. The output is a discretized latent codeword, which is then mapped back into continuous latent space using a custom-designed reverse Gaussian bin-discretization function.

3) Long Short-Term Memory: As a classical machine learning baseline, an LSTM network was implemented to model the prior distribution of the latent space [24]. The latent space dimensions were treated as elements in a sequence. The LSTM architecture consisted of a single layer with 256 neurons trained for 100 epochs. During training, the model received random noise as the initial input and employed the teacher forcing strategy, where the true output from the previous time step was fed as input to the current step [28], [29]. The LSTM was trained to predict the next latent dimension in the sequence. The training objective minimized the Mean Squared Error (MSE) between the predicted latent dimension and the true latent dimension.

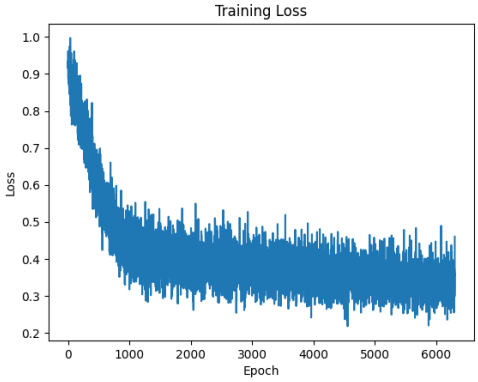

Fig. 7. LSTM MSE Training Losses

D. Model Comparison

A comprehensive comparative analysis was conducted to evaluate the performance of the trained QCBM, QGAN, and LSTM models in capturing and sampling from the VQ-VAE's latent space distribution. Samples were generated from each trained model, and their distributions were compared against the original VQ-VAE codebook data, which served as the ground truth. These samples were concatenated into arrays that served as the basis for comparison.

1) T-SNE Visualization: T-distributed Stochastic Neighbor Embedding (t-SNE) was used to visualize the high-dimensional latent spaces sampled by each model in a lower, two-dimensional space [27]. A perplexity value of 100 was used for the t-SNE projection. To ensure a relativistic comparison, the t-SNE model was fitted on the combined dataset consisting of samples from all three generative models.

2) PCA Visualization: Principal Component Analysis (PCA) was also employed for dimensionality reduction and visualization of the latent spaces [8]. Similar to the t-SNE analysis, PCA was performed on the concatenated dataset of samples from all models. The projection onto the first

two principal components provides a linear perspective on the structure and spread of the learned and true distributions, complementing the non-linear view provided by t-SNE.

3) Average Minimum Distances: A quantitative comparison was performed by calculating the average minimum Euclidean distance between the samples generated by each model and the vectors in the original VQ-VAE codebook [25], [26]. For each generated sample, the Euclidean distance to every vector in the codebook was computed, and the minimum distance was identified. The average of these minimum distances across all generated samples for a given model provides a numerical measure of how closely the generated distribution approximates the true distribution.

4) Nearest Neighbor Analysis: A Nearest Neighbor analysis provided a discretized evaluation of model performance [25], [26]. For each vector in the original VQ-VAE codebook, the closest generated sample among all samples from all three models (QCBM, QGAN, LSTM) was identified. This analysis determined which model's samples were most frequently the nearest neighbors to the true latent space vectors, offering insight into which model best covers the high-density regions of the true distribution.

5) Models Distance Distribution: Finally, the Nearest Neighbour model was employed again to compute the minimal distance between each codeword in the original VQ-VAE codebook and the corresponding closest sample generated by each model independently [25], [26]. This allowed for plotting and comparing the distance distributions, offering further quantitative insight into each model's ability to reproduce the original latent space accurately.

V. Results and Interpretation

This section presents the experimental results obtained from the developed computational framework and the subsequent analysis of the generative models' performance. The findings are interpreted in the context of the research objectives, emphasizing their implications for the efficient modeling and generation of fluid dynamics data using classical and quantum approaches.

A. LBM Fluid Simulator Performance and Data Generation

The custom GPU-accelerated Lattice Boltzmann Method (LBM) simulator developed for this study demonstrated robust performance and the capability to generate physically accurate fluid dynamics data for both 2D and 3D geometries. Validation against established test cases confirmed the simulator's fidelity in reproducing expected flow phenomena, including characteristics such as vortex shedding at appropriate Reynolds numbers. Visualizations of 2D flow around an airplane profile at Re=1000 and 3D flow around a sphere qualitatively demonstrated the simulator's ability to capture complex turbulent structures and its functionality in three dimensions. These results validated the simulator as a reliable tool for generating the required fluid dynamics datasets. The efficiency gained through GPU acceleration was critical for generating the large datasets necessary for training the deep learning and quantum models within a reasonable timeframe. The primary dataset utilized in this research consisted of 2D vorticity profiles of flow around a cylinder at Re=500 and 0.1 Mach inlet velocity, providing the necessary data for learning the compressed latent space. Figure 1 is an example of one of these vorticity snapshots that were used in training.

B. VQ-VAE Latent Space Learning and Characteristics

The Vector Quantized Variational Autoencoder (VQ-VAE) was successfully trained on the generated 2D fluid vorticity data to learn a compressed, discrete latent space representation [12]. The training process exhibited stable convergence, as indicated by the behavior of the reconstruction and quantization loss curves. The rapid and smooth convergence of the reconstruction loss demonstrates the VQ-VAE's effectiveness in learning to encode the high-dimensional vorticity fields into the latent space and decode them back with high fidelity.

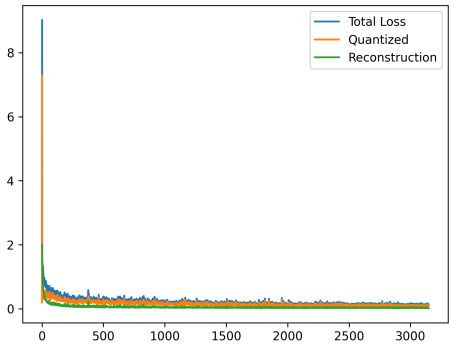

Fig. 8. VQ-VAE Training Losses

The convergence of the quantization loss confirms that the model successfully formed a discrete codebook of embeddings that capture the underlying structure of the fluid flow data. Analysis of the learned discrete latent space, visualized through dimensionality reduction techniques such as t-SNE [27], revealed a structured distribution. The t-SNE projection of the VQ-VAE codebook indicated that the model effectively mapped the complex, high-dimensional vorticity data to a lower-dimensional discrete space [12]. This projection showed a distribution that appeared somewhat clustered but with a discernible overall structure, resembling a discretized, multimodal distribution.

This suggests that the VQ-VAE successfully reduced dimensionality while preserving meaningful variations in the flow patterns, represented by distinct codewords. This successful compression and discretization of the fluid dynamics data into a meaningful latent space was a critical step, providing a suitable target distribution for the subsequent classical and quantum generative modeling experiments.

C. Comparative Analysis of Generative Models

The primary objective of this research was to compare the performance of classical and quantum generative models in

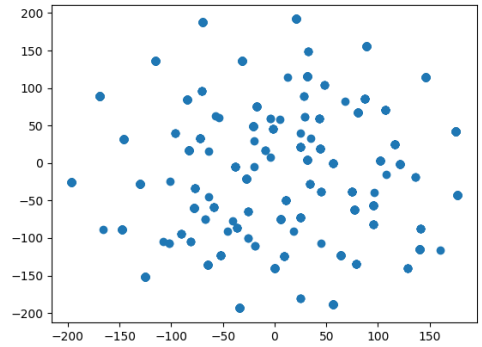

Fig. 9. VQ-VAE Latent Space T-SNE Visualization

learning and sampling from the discrete latent space of fluid vorticity data. A comprehensive analysis was conducted using both visualization techniques and quantitative metrics to assess how well the trained QCBM, QQAN, and LSTM models captured the empirical distribution of the VQ-VAE codebook.

1) Latent Space Visualization (t-SNE and PCA): Visualization of the sampled latent spaces using dimensionality reduction techniques provided initial qualitative insights into the models' performance. The t-SNE projection of samples from the QCBM, QGAN, and the LSTM revealed distinct patterns when projected into two dimensions.

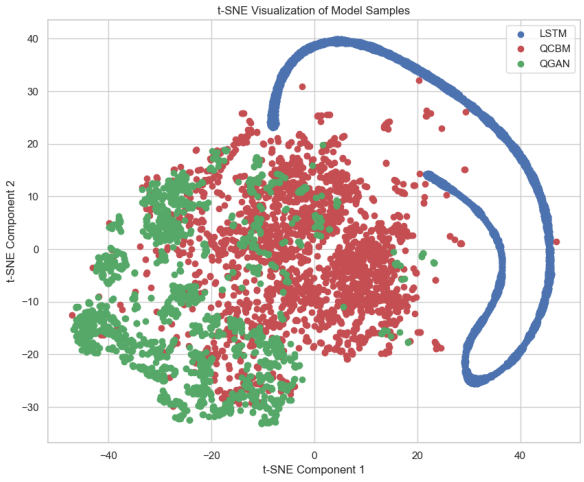

Fig. 10. T-SNE Comparative Visualization

The samples generated by the classical LSTM model exhibited a relatively constrained, almost linear-like structure in the 2D projection space. This suggests that the LSTM, when applied to model the sequence of latent dimensions, may have struggled to fully capture the potentially non-linear and complex correlations present in the multi-dimensional latent space distribution. In contrast, the quantum models (QCBM and QGAN) produced samples that occupied a broader region of the projected latent space, indicating potentially greater expressiveness in capturing the latent distribution [13]. The QCBM samples tended to form a more concentrated, Gaussian-like cluster in the center of the projection. The QGAN samples, while also covering a wider area than the LSTM, appeared more fragmented and clustered in different regions of the projected space. This visual difference between the QCBM and QGAN outputs suggests that their distinct architectures and training approaches led to different strategies for modeling the latent distribution. The PCA visualization of the latent spaces further supported these observations [8].

Fig. 11. PCA Comparative Visualization

The PCA results clearly showed the LSTM samples aligning predominantly along a linear manifold defined by the first two principal components. The QCBM samples again formed a more centralized cluster, consistent with a distribution that is well-approximated by its principal components. The QGAN samples were more scattered, with some points appearing further from the central mass, reinforcing the idea of a more fragmented or multi-modal distribution capture. These visualizations collectively suggest that the quantum models were able to learn more complex and potentially higher-dimensional relationships within the latent space compared to the classical LSTM baseline.

2) Quantitative Performance Metrics: To provide a more rigorous evaluation, quantitative metrics were computed to compare the generative models' ability to reproduce the empirical VQ-VAE codebook distribution. The average minimum Euclidean distance between generated samples and the original codebook vectors served as a key performance indicator. Results showed that the QCBM achieved the lowest average minimum distance, indicating that its generated samples were, on average, closest to the true latent space codewords. The QGAN had a higher average minimum distance than the QCBM but was significantly lower than the LSTM. The classical LSTM exhibited the largest average minimum distance, quantitatively

confirming the visual assessment that its generated distribution was less aligned with the true latent space.

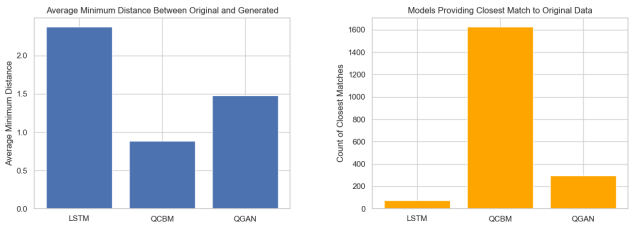

Fig. 12. Quantitative Analysis

The Nearest Neighbor analysis provided a discretized perspective on which model best covered the high-density regions of the true distribution. This analysis revealed that samples generated by the QCBM were the nearest neighbors to the vast majority of the original VQ-VAE codebook vectors. The QGAN had a considerably smaller number of samples identified as nearest neighbors, and the LSTM had very few samples that were the closest to any codebook vector. This finding strongly suggests that the QCBM was most effective at generating samples that fall within the vicinity of the actual data points in the latent space. Analysis of the distribution of minimum distances between codebook vectors and the closest generated samples from each model provided further insight. Histograms of these distance distributions showed that the histogram for the QCBM had a sharp peak at a low distance value, indicating that most codebook vectors were very close to a generated sample. The QGAN's distance distribution was broader and shifted towards higher distances compared to the QCBM, while the LSTM's distribution was widely spread across much larger distances. These distributions quantitatively illustrate the superior ability of the QCBM, followed by the QGAN, to generate samples that are proximate to the true latent space data points, relative to the LSTM baseline under the experimental conditions.

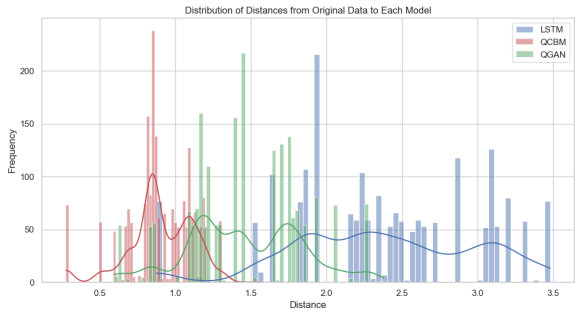

Fig. 13. Minimum Distances Distribution Visualization

D. Interpretation of Findings

The comparative analysis yielded several significant findings regarding the ability of classical and quantum generative models to capture the empirical distribution of the VQ-VAE latent space. Firstly, the results consistently demonstrate that the quantum generative models (QCBM and QGAN), within the scope of these experiments, outperformed the classical LSTM baseline in modeling and sampling from the discrete latent space of fluid vorticity data. This is a notable outcome, suggesting that even with current quantum simulation capabilities and the specific architectures and training parameters used, quantum approaches can offer advantages for modeling complex distributions in scientific data compared to a widely used classical sequence model [13]. This supports the potential utility of quantum machine learning for tasks in scientific computing domains. Secondly, among the quantum models evaluated, the QCBM, which modeled the independent distribution of each latent dimension and was trained for 100 iterations, achieved the best overall performance according to the quantitative metrics. This result was observed despite the QGAN being designed with an architecture intended to capture conditional dependencies and trained for 2 epochs on the whole data. The disparity in training duration between the QCBM (100 iterations) and the QGAN (2 epochs) is a significant experimental factor that could influence their relative performance. It is also possible that the discrete latent space learned by the VQ-VAE for this specific dataset has dimensions that are largely independent, which would favor the QCBM's approach. The observed performance difference between the quantum models and the classical LSTM highlights the potential of quantum circuits for capturing intricate patterns in the latent space that may be challenging for classical autoregressive models like the LSTM when applied to this type of data representation. While LSTMs are powerful for sequential data, the structure of the VQ-VAE latent space, although treated sequentially for the LSTM, may not align optimally with the LSTM's inherent inductive biases compared to the distributional modeling capabilities of the quantum circuits. These findings underscore the potential of integrating quantum machine learning techniques with classical dimensionality reduction methods for analyzing and generating complex scientific data. While challenges related to quantum hardware accessibility and scalability remain, this research provides empirical evidence that current quantum approaches, even in simulation and with specific training parameters, can offer valuable capabilities for generative modeling in scientific domains like fluid dynamics. The results warrant further investigation into the factors influencing the performance of different quantum generative architectures and the impact of training methodologies and data characteristics on their ability to model complex scientific data distributions.

VI. Discussion

The results presented in Section V provide significant insights into the application of classical and quantum generative models for learning and sampling from the latent space of fluid dynamics data. The successful development and validation of the GPU-accelerated LBM simulator and the VQ-VAE for generating and compressing fluid vorticity data

established a robust foundation for the comparative analysis of generative models. A key finding of this research is the consistent outperformance of the quantum generative models compared to the classical LSTM baseline in modeling the discrete latent space distribution. This result is particularly noteworthy as it suggests that, even when simulated on classical hardware, quantum-inspired or quantum-based approaches can offer advantages for capturing the complex patterns inherent in scientific data representations, such as the latent space derived from fluid vorticity. This finding supports the growing interest in quantum machine learning for tasks beyond those where a quantum speedup is theoretically guaranteed, highlighting their potential in complex data generation and analysis within scientific computing domains. The ability of quantum circuits to naturally represent and manipulate high-dimensional probability distributions likely contributes to their superior performance over a classical autoregressive model like LSTM, which may be less suited to capturing the full, potentially non-sequential, correlations within the VQ-VAE latent space. The comparative performance between the two quantum models, where the QCBM (modeling independent latent dimensions) outperformed the QGAN (designed for conditional dependencies), presents an interesting point for discussion. This outcome could imply that the VQ-VAE, through its training process, effectively decorrelated the dimensions of the latent space, making independent modeling a highly effective strategy. Alternatively, it might indicate challenges in the training stability or scalability of the QGAN implementation, particularly given the constraints imposed by simulating multi-qubit entanglement and the patch-based approach necessary to handle the latent space dimensionality on classical simulators. Future work could involve a more detailed analysis of the latent space's statistical properties to assess the degree of independence between dimensions and further optimization of the QGAN training process. The implications of these findings are significant for the field of CFD [1], [2] and scientific computing. The ability to accurately model and sample from a compressed latent space of fluid flow data [12] opens avenues for more efficient data augmentation, generation of synthetic training data for downstream tasks (e.g., training surrogate models), and potentially accelerating design optimization loops by rapidly generating diverse flow configurations. Furthermore, demonstrating the potential of quantum generative models in this context motivates continued research into their application for other complex scientific datasets and the development of more powerful quantum hardware and algorithms tailored for such tasks [13].

VII. Limitations and Future Work

Despite these promising results, several limitations must be acknowledged. The quantum models were evaluated using classical simulations, which do not fully capture the nuances and potential noise of actual quantum hardware [13]. The scalability of these quantum approaches to significantly larger latent spaces or higher-dimensional CFD problems on near-term quantum devices remains a challenge. The study focused on a specific 2D fluid flow scenario (cylinder at Re=500), and the generalizability of the findings to other flow regimes, geometries, or 3D simulations needs further investigation. Additionally, the properties of the VQ-VAE latent space itself, including the optimal number of dimensions and codebook size, could influence the performance of the generative models [12]. Future research should explore the application of these techniques to more complex and higher-dimensional fluid dynamics problems, potentially leveraging advancements in quantum hardware and fault-tolerant quantum algorithms as they become available. Investigating alternative quantum generative model architectures and training strategies, particularly for capturing complex dependencies in latent spaces, is also a promising direction. Furthermore, exploring the use of the generated latent space samples for specific downstream CFD tasks, such as training surrogate models for accelerated prediction or optimizing flow control strategies, would demonstrate the practical utility of this approach. Finally, analyzing the mathematical and physical properties encoded within the learned latent space and how they are captured by the different generative models could provide deeper scientific insights.

VIII. Conclusion

This research successfully investigated the application of classical and quantum generative models to the challenging problem of efficiently modeling and generating complex fluid dynamics data. By developing a robust GPU-accelerated Lattice Boltzmann Method (LBM) simulator [17], [18], we generated high-fidelity fluid vorticity data, which was then effectively compressed into a discrete, low-dimensional latent space using a Vector Quantized Variational Autoencoder (VQ-VAE) [12]. This established a foundation for exploring advanced generative modeling techniques on a more manageable data representation. The core contribution of this work lies in the comprehensive comparative analysis of a classical Long Short-Term Memory (LSTM) network [24] against two prominent quantum generative models, the Quantum Circuit Born Machine (QCBM) [14] and the Quantum Generative Adversarial Network (QGAN) [15], [16], for learning and sampling from the empirical distribution of this discrete latent space. Our results, obtained under specific experimental conditions and training parameters, provide empirical evidence that the quantum generative models, as implemented and simulated, consistently demonstrated a stronger capability in capturing the latent space distribution compared to the classical LSTM baseline. This was evidenced by quantitative metrics including average minimum distance and nearest neighbor analysis, complemented by qualitative visualizations (t-SNE [27], PCA [8]). Specifically, the QCBM, employing an independent modeling approach for each latent dimension and trained for 100 iterations, achieved the most favorable performance metrics among the models evaluated. While the QGAN, designed with an architecture intended to capture conditional dependencies and trained for 2 epochs, also showed improved performance relative to the LSTM, its results suggest that factors such as the training duration or the characteristics of the latent

space may have influenced its performance relative to the QCBM. These findings are of significant value to the scientific computing [1], [2] and quantum machine learning communities [13]. They empirically demonstrate the potential of current quantum computing approaches, even when evaluated through classical simulation and within the constraints of near-term hardware capabilities and training methodologies, to offer advantages over established classical techniques for complex data generative tasks in scientific domains like fluid dynamics. This research validates the utility of integrating quantum machine learning with classical dimensionality reduction methods as a promising avenue for handling and generating intricate scientific data. In conclusion, this study represents a significant step towards leveraging emerging quantum technologies for advanced scientific data analysis. By showcasing the comparatively stronger performance of quantum generative models in learning the latent structure of fluid dynamics under the specified experimental conditions, this work opens new possibilities for more efficient data representation, synthesis, and potentially accelerated discovery in CFD and other computationally intensive fields [1], [2], [13]. The successful methodology and promising results lay the groundwork for future investigations into scalable quantum algorithms, the impact of training methodologies, and their application to increasingly complex scientific challenges on both simulated and future quantum hardware.

ACKNOWLEDGMENT

The author thanks Dr. Fouad Mohammed Abbou for guidance and discussions throughout the research. Dr. Lamiae Bouanane and Professor Mouna Kettani are acknowledged for their roles in supporting quantum computing research within the School Of Science and Engineering. The conceptualization, technical development, and preparation of the manuscript were carried out by the first author.

REFERENCES

[1] M. N. Dhaubhadel, "Review: CFD Applications in the Automotive Industry," *J. Fluids Eng.*, vol. 118, no. 4, pp. 647–653, Dec. 1996.

[2] M. Mani and A. J. Dorgan, "A perspective on the state of aerospace CFD technology," *Annu. Rev. Fluid Mech.*, vol. 55, pp. 431–457, Jan. 2023.

[3] R. McConkey, E. Yee, and F.-S. Lien, "A curated dataset for data-driven turbulence modelling," *Sci. Data*, vol. 8, Art. 255, Aug. 2021.

[4] S. Bond-Taylor *et al.*, "Deep generative modelling: A comparative review of VAEs, GANs, normalizing flows, energy-based and autoregressive models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 11, pp. 7327–7347, Nov. 2021.

[5] I. J. Goodfellow *et al.*, "Generative adversarial nets," in *Adv. Neural Inf. Process. Syst.*, vol. 27, 2014.

[6] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," Dec. 2013.

[7] P. Baldi, "Autoencoders, unsupervised learning, and deep architectures," in *Proc. ICML Workshop Unsupervised Transfer Learn.*, JMLR Workshop Conf. Proc., 2012.

[8] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006.

[9] P. Vincent *et al.*, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, no. 12, 2010.

[10] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *Proc. Int. Conf. Mach. Learn.*, 2014.

[11] I. Higgins *et al.*, "β-VAE: Learning basic visual concepts with a constrained variational framework," in *Proc. Int. Conf. Learn. Represent.*, 2017.

[12] A. van den Oord and O. Vinyals, "Neural discrete representation learning," in *Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.

[13] M. Hibat-Allah, M. Mauri, J. Carrasquilla, and A. Perdomo-Ortiz, "A framework for demonstrating practical quantum advantage: comparing quantum against classical generative models," *Commun. Phys.*, vol. 7, pp. 1–9, 2024, doi:10.1038/s42005-024-01552-6.

[14] M. Benedetti *et al.*, "A generative modeling approach for benchmarking and training shallow quantum circuits," *NPJ Quantum Inf.*, vol. 5, no. 1, p. 45, 2019.

[15] S. Lloyd and C. Weedbrook, "Quantum generative adversarial learning," *Phys. Rev. Lett.*, vol. 121, no. 4, p. 040502, Jul. 2018.

[16] P.-L. Dallaire-Demers and N. Killoran, "Quantum generative adversarial networks," *Phys. Rev. A*, vol. 98, no. 1, p. 012324, Jul. 2018.

[17] S. Chen and G. D. Doolen, "Lattice Boltzmann method for fluid flows," *Annu. Rev. Fluid Mech.*, vol. 30, no. 1, pp. 329–364, Jan. 1998.

[18] J. Latt *et al.*, "Multi-GPU Programming with Standard Parallel C++, Part 2," *NVIDIA Developer Blog*, Apr. 18, 2022. [Online]. Available: https://developer.nvidia.com/blog/multi-gpu-programming-with-standard-parallel-cpp-part-2/ (accessed May 3, 2025).

[19] Q. Zou and X. He, "On pressure and velocity boundary conditions for the lattice Boltzmann BGK model," *Phys. Fluids*, vol. 9, no. 6, pp. 1591–1598, Jun. 1997.

[20] M. Bouzidi, M. Firdaouss, and P. Lallemand, "Momentum transfer of a Boltzmann-lattice fluid with boundaries," *Phys. Fluids*, vol. 13, no. 11, pp. 3452–3459, Nov. 2001.

[21] C. L. M. H. Navier, "Mémoire sur les lois du mouvement des fluides," *Mém. R. Acad. Sci. Inst. France*, vol. 6, pp. 389–440, 1823.

[22] G. G. Stokes, "On the theories of the internal friction of fluids in motion, and of the equilibrium and motion of elastic solids," 2007.

[23] O. Reynolds, "An Experimental Investigation of the Circumstances which Determine whether the Motion of Water shall be Direct or Sinuous," *Phil. Trans. R. Soc. Lond.*, vol. 174, pp. 935–982, 1883.

[24] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[25] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *J. Am. Stat. Assoc.*, vol. 112, no. 518, pp. 859–877, 2017.

[26] A. Gretton *et al.*, "A kernel two-sample test," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 723–773, Mar. 2012.

[27] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, 2008.

[28] A. M. Lamb *et al.*, "Professor forcing: A new algorithm for training recurrent networks," in *Adv. Neural Inf. Process. Syst.*, vol. 29, 2016.

[29] Y. Bengio *et al.*, "Scheduled sampling for sequence prediction with recurrent neural networks," in *Adv. Neural Inf. Process. Syst.*, vol. 28, 2015.