Universiteit Leiden
The Netherlands

# Bachelor Data Science and Artificial Intelligence

Comparison of Classical, Hybrid Quantum-Classical, and Quantum Generative Models on the Low-Dimensional Bars and Stripes Dataset

Gabriela Czapska

Supervisors:
Evert van Nieuwenburg & Felix Frohnert

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

01/07/2026

**Abstract**

This thesis investigates classical, hybrid quantum-classical, and quantum approaches to generative modelling using the low-dimensional $3 \times 3$ Bars and Stripes dataset. Three models are compared: a classical Wasserstein Generative Adversarial Network with Gradient Penalty (WGAN-GP), a hybrid quantum-classical WGAN-GP, and a Quantum Circuit Born Machine (QCBM). The models are evaluated using *support validity*, *support coverage*, and *Total Variation (TV) distance* in order to assess the structural validity and diversity of generated samples within each batch, as well as the similarity between the learned and target probability distributions.

The WGAN-GP-based models are trained using adversarial optimisation, whereas the QCBM optimises the probability distribution encoded in the quantum circuit by minimising the kernel-based Maximum Mean Discrepancy (MMD) loss. The quantum models are implemented with PennyLane using the `default.qubit` state-vector simulator. Therefore, the results should be interpreted as a preliminary assessment of quantum approaches under classical simulation against a classical baseline, rather than as evidence for or against quantum advantage.

The QCBM achieved the strongest overall performance, with the highest validity ratio, complete support coverage, and the lowest Total Variation distance. The classical WGAN-GP consistently outperformed the hybrid quantum-classical WGAN-GP, suggesting that the inclusion of a parametrised quantum circuit (PQC) within an adversarial generator did not provide a measurable improvement for this task. Overall, the results show that quantum and hybrid quantum-classical generative models can learn non-trivial discrete probability distributions but that their effectiveness depends strongly on the selected quantum model, ansatz, and training objective. Future research should investigate whether these findings persist for larger datasets, more complex distributions, and implementations on real quantum hardware.

## Acknowledgments

I would like to sincerely thank my supervisor, Evert van Nieuwenburg, for his trust and support throughout the writing process of this thesis. This work has helped me develop both professionally and personally.

# Contents

# Glossary

**Ansatz** An educated guess for the structure of a parameterised quantum circuit used to represent a candidate solution to a quantum problem. It is defined by the arrangement of gates, the number of qubits, and entanglement patterns.

**Bloch Sphere** A geometric representation of a single-qubit state as a point on a unit sphere. It is used to visualise superposition and quantum operations.

**Born Rule** A mathematical rule that determines the probability of obtaining a particular measurement outcome from a quantum state. The probability that a measurement of the quantum state $|\psi\rangle$ yields the basis state $|x\rangle$ is given by $P(x) = |\langle x|\psi\rangle|^2$, where $\langle x|\psi\rangle$ corresponds to the probability amplitude associated with outcome $x$.

**Classical Quantum Simulation** A process of simulating quantum behaviour on classical hardware using mathematical models.

**Computational (Z-)Basis** The standard basis used to describe and measure qubits. It corresponds to the Z-basis on the Bloch sphere, such that upon projective measurement in this basis, a quantum state can collapse to either $|0\rangle$ or $|1\rangle$.

**Entanglement** A quantum phenomenon in which two or more qubits are described co-dependently, regardless of the physical distance between them. Measuring one qubit in an entangled state constrains the possible outcomes of measurements on the other correlated qubits.

**Fault-Tolerant Quantum Computing (FTQC)** A paradigm of large-scale quantum computing in which quantum error correction enables reliable computation despite the presence of physical errors.

**Hilbert space** An abstract vector space used to represent quantum states. It is equipped with an inner product, which allows vector overlaps and magnitudes to be measured, and may have either finite or infinite dimensionality.

**Noisy Intermediate-Scale Quantum (NISQ)** A class of quantum hardware that is limited in size and characterised by high error rates and limited error correction.

**Parametrised Quantum Circuit (PQC)** A quantum circuit containing tunable parameters, typically corresponding to the rotation angles of parametrised quantum gates, with its structure defined by an ansatz. Circuit parameters can be adjusted to represent different quantum states in order to perform a variety of computational tasks.

**Projective Measurement** A quantum measurement that collapses a quantum state into a discrete classical outcome. The probability of obtaining a specific outcome is determined by the Born rule.

**Pure Quantum State** A quantum state that contains complete information about a quantum system and is represented by a state vector $|\psi\rangle$.

**Quantum Circuit Born Machine (QCBM)** An unsupervised generative model that uses a parametrised quantum circuit to prepare a quantum state that encodes a probability distribution and generates samples from it through quantum measurement.

**Quantum Computing** A field of computer science that uses the principles of quantum mechanics to process information and perform computations.

**Quantum Decoherence** A process in which a quantum system loses its quantum properties due to interactions with the environment. The underlying cause is the entanglement of a fragile qubit state with its surroundings, which reduces its ability to maintain superposition and results in more classical behaviour.

**Quantum Error** An unwanted change in quantum information encoded in qubits.

**Quantum Gate** A building block of a quantum circuit that applies operations to one or more qubits in order to manipulate their quantum state.

**Quantum Information** Any information encoded within a quantum system.

**Quantum Noise** A source of errors in a quantum state that can be caused by unwanted interactions with the environment.

**Quantum State Collapse** A process in which a quantum state existing in a superposition collapses into a single, discrete outcome upon measurement.

**Qubit** A basic unit of quantum information that can exist in the superposition of computational basis states, as visualised on the Bloch sphere, and become entangled with other qubits.

**State Vector ($|\psi\rangle$)** A mathematical representation of a quantum system that describes its state at a given point in time.

**Superposition** A fundamental property of a quantum system in which a quantum state is represented as a linear combination of multiple states. For a single qubit, it is expressed as $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$, where $|0\rangle$ and $|1\rangle$ are the computational basis states, and the coefficients $\alpha$ and $\beta$ are complex probability amplitudes associated with each basis state.

**Unitary** A property of a matrix representing a quantum operation, such as a quantum gate. Unitary operations are reversible and preserve the norm of quantum states, ensuring that probabilities of all measurement outcomes sum to 1.

**Variational Quantum Circuit (VQC)** A hybrid quantum-classical framework that uses a classical optimisation algorithm to iteratively update the tunable parameters of a parametrised quantum circuit (PQC).

# 1 Introduction

The intersection of quantum computing and machine learning has given rise to the interdisciplinary field of Quantum Machine Learning (QML) [FSD+26]. By integrating the principles of quantum mechanics with Artificial Intelligence (AI), QML seeks to exploit quantum phenomena, such as superposition and entanglement, for potential computational advantages over the classical methods. The current state of quantum computing is referred to as the Noisy Intermediate-Scale Quantum (NISQ) era, a term coined by John Preskill [Pre18] to characterise near-term quantum hardware that is limited in size and characterised by high error rates. Interaction with the environment can introduce noise, such as heat, to the quantum system that can lead to the emergence of quantum errors. Due to the fragile nature of qubits, these errors may corrupt the information encoded within a quantum circuit or cause quantum decoherence, resulting in a more classical behaviour.
As quantum systems scale, the error propagation escalates, and maintaining reliable qubit interactions becomes increasingly difficult. This imposes size limitations that consequently restrict the complexity of problems that quantum circuits can encode. Progressing towards a post-NISQ era therefore requires the development of fault-tolerant quantum systems capable of operating with thousands to millions of qubits.

## 1.1 Quantum Advantage

One of the central research directions within quantum computing is the investigation of quantum advantage [HBE+25], which examines whether quantum computers can outperform classical computers on computational tasks, particularly those currently intractable, such as integer factorisation.
Two conditions must be satisfied for an information-processing task solved with quantum hardware to establish quantum advantage. The first is that the correctness of the quantum output must be rigorously validated. Given the novelty of the paradigm, this is a particularly critical requirement, since direct comparison can be difficult for problems where no classical methods are established. Alternative validation strategies include verifying individual steps of the calculation to assess the correctness of the entire computation. However, since fault-tolerant quantum computing (FTQC) is not yet available, any claim of quantum advantage should be treated as a hypothesis to be challenged and iteratively verified, rather than as a conclusion established at a single point in time. The second condition is a provable quantum separation in algorithmic performance, defined as a gap in efficiency, cost-effectiveness, or accuracy compared to the best-performing classical methods, or supported by theoretical assumptions in complexity [OL25]. Such an advantage should remain robust over time and withstand verification against both existing and emerging classical techniques. Sampling problems represent one of the most promising near-term application domains for quantum advantage. Their inherent randomness closely corresponds to the non-deterministic outcomes of quantum mechanics, with quantum circuits encoding probability distributions that, upon measurement, produce deterministic samples. This positions quantum generative modelling as a well-motivated candidate for investigating quantum advantage, especially that it can tackle more complex problems compared to classification tasks.
A definitive proof of quantum advantage can only be established on deployable, fault-tolerant quantum hardware. While current quantum hardware remains noisy and, due to long queue times, difficult to access, classical simulation of quantum systems, as employed in this study, circumvents these limitations. Simulation, however, serves only as a crucial preliminary assessment of the

algorithmic prototypes, enabling their debugging and testing without the cost of using real quantum hardware. Still, the exponential memory requirements of classically simulating quantum systems ($2^n$ for an $n$-qubit system) restrict such simulations to small-scale, low-dimensional, few-qubit problems. Rather than exploiting genuine quantum effects, these simulations rely on classical mathematical models of quantum behaviour. This study is therefore positioned as a classical simulation-based investigation of quantum approaches to generative modelling and is guided by the following research question:

> *How does the performance of a classical Wasserstein Generative Adversarial Network with Gradient Penalty (WGAN-GP) compare to that of a hybrid quantum-classical WGAN-GP and a Quantum Circuit Born Machine (QCBM) in generative modelling, as evaluated using the validity ratio, coverage ratio, and Total Variation distance?*

## 1.2 Thesis overview

This thesis was conducted within the Leiden Institute of Advanced Computer Science (LIACS) at Leiden University and is structured as follows. Section 2 introduces the foundational concepts needed to understand the models developed in this work, including classical adversarial generative modelling, the Wasserstein distance, the 1-Lipschitz constraint, gradient penalty, and key quantum computing concepts such as superposition, entanglement, parametrised quantum circuits (PQCs), and the Born rule. Section 3 reviews relevant literature on classical, quantum, and hybrid quantum-classical generative models, with a focus on motivating the choice of architectures investigated in this study. Section 4 describes the experimental setup, including the Bars and Stripes dataset, model architectures, training procedures, selected hyperparameters, implementation details, computational resources, and evaluation metrics. Section 5 presents the results for all three models and compares their performance using the selected metrics. Section 6 analyses the results, discusses limitations, and considers possible explanations for the observed performance differences. Finally, Section 7 summarises the findings, addresses the research question, and proposes directions for future work.

# 2 Theoretical Background

## 2.1 Classical Generative Modelling

### 2.1.1 Generative Adversarial Networks

Generative Adversarial Networks (GANs) consist of two competing networks: a generator and a discriminator. They are trained in a framework in which the gain of one model causes the loss of the other. The primary goal of the discriminator is to minimise the accumulated cost over all samples, as measured by the Binary Cross-Entropy (BCE) loss computed on individual data points. A low cost value indicates that the discriminator successfully distinguishes between real and generated samples. At the same time, the generator aims to increase the discriminator's classification error, as reflected in a higher BCE loss per generated sample, by producing samples that closely resemble the target data, thus increasing the chance of deceiving the discriminator. This dynamic between

the networks is referred to as a minimax game. The Binary Cross-Entropy (BCE) loss function is defined as

$$\mathcal{L}_{\text{BCE}} = -\left(y \cdot \log(x) + (1-y) \cdot \log(1-x)\right),$$

where $y$ is the true label of the sample, and $x$ is the discriminator's predicted probability that the sample is real. For real samples ($y = 1$), the loss reduces to $-\log(x)$, evaluating how well the real data is identified as real. For generated samples ($y = 0$), the loss reduces to $-\log(1-x)$, evaluating how well the generated data is identified as generated.
The generator responsible for producing synthetic data is guided by the discriminator that evaluates it. To ensure that the training signal provided to the generator remains informative, the discriminator is typically trained for more iterations than the generator [GPAM+14]; in this implementation, it is trained five times more often. This allows the discriminator to maintain a slight advantage during training and provide meaningful feedback to the generator. Such guidance is particularly important given the complexity of the generator's task, which involves transforming latent noise vectors into complete data patterns, whereas the discriminator only produces scalar outputs.
In standard GANs, the discriminator employs a sigmoid activation function to convert its output score into the probability that a given sample $x$ is real, defined as

$$\sigma(x) = \frac{1}{1+e^{-x}}.$$

A higher probability value indicates a more realistic input, and the resulting probability is later used by the BCE loss to assess classification performance.
BCE, however, can result in vanishing gradients for generated samples that receive scores close to 0 and therefore fail to deceive the discriminator, while amplifying the signal for synthetic samples that receive scores close to 1 and more closely resemble the target patterns. Such a confident discriminator, with limited intermediate scores to guide the learning process, reduces the task to binary classification and can lead to mode collapse. Mode collapse occurs when the generator is encouraged to produce only a subset of target patterns, thus reducing the diversity of data structures produced by the generator and preventing it from reflecting the full spectrum of the target probability distribution.
The underlying cause of these limitations is that the BCE loss measures the confidence and correctness of the classification, rather than the distance between probability distributions. As a result, it neglects the repetitiveness and structural diversity of the generated samples. Once the discriminator establishes a clear and confident separation between the real and generated distributions, the generator receives progressively less informative gradients, which can hinder learning and encourage the repeated generation of only those patterns that would most likely deceive the discriminator.

### 2.1.2 Wasserstein Generative Adversarial Network

A Wasserstein GAN (WGAN) is an extension of the standard GAN framework in which the discriminator is replaced by a critic that operates without a sigmoid activation function. Instead of bounding the outputs within the interval $[0, 1]$, the critic assigns real-valued scores to the samples, with larger scores indicating that a sample is judged to be more realistic. These critic scores,

denoted by $C$ in Equation (1), are used to estimate the Wasserstein distance between the real and generated probability distributions, replacing the BCE loss used in the original GAN formulation:

$$W_{\text{est}} = \mathbb{E}[C(x_{\text{real}})] - \mathbb{E}[C(x_{\text{generated}})]. \tag{1}$$

In practice, and as reflected in the code implementation, this objective is computed as the difference between the mean critic scores assigned to real and generated samples, where $\mathbb{E}$ denotes the average over the corresponding scores. It therefore evaluates similarity at the distribution level, since the real-valued critic scores provide a numerical measure of how generated samples differ from real ones, allowing structural differences between generated and target patterns to be quantified instead of reducing the task to a binary classification. Conceptually, it accounts for both the amount of probability mass and the distance over which it must be moved between two probability distributions. As in the standard GAN framework, the critic and generator optimise opposing objectives. The critic seeks to maximise the Wasserstein estimate by assigning higher scores to real samples and lower scores to generated samples, thus increasing the estimated distance between the two distributions. By contrast, the generator seeks to minimise this estimate by producing samples that receive increasingly higher scores from the critic, reducing the distance between the generated and target distributions and making the generated samples increasingly difficult for the critic to distinguish from the real ones.

The Wasserstein objective provides a continuous and smooth measure of the difference between the generated and target distributions, allowing meaningful gradients to be provided even when their overlap is minimal. This allows the generator to receive an informative training signal even when the samples it generates are assigned low critic scores. Consequently, the Wasserstein loss mitigates the vanishing gradient problem and reduces the risk of mode collapse.

However, for the critic to operate correctly, it must satisfy the 1-Lipschitz continuity constraint, formulated as

$$|C(x_1) - C(x_2)| \leq \|x_1 - x_2\|_2, \quad \text{or equivalently} \quad \|\nabla_x C(x)\|_2 \leq 1$$

This constraint limits the change in the critic's output to at most linear growth with respect to changes in the input. Since the inputs $x_1$ and $x_2$ are vectors, their difference is measured using the $\ell_2$ norm, while the difference between the scalar critic outputs is measured using an absolute value. Equivalently, the constraint bounds the $\ell_2$ norm of the critic gradients by one. As a result, the critic is prevented from responding too strongly to small input changes, which stabilises the optimisation process and prevents exploding gradients.

### 2.1.3 Gradient Penalty

In the standard WGAN framework, the 1-Lipschitz constraint is enforced through weight clipping, which restricts the critic weights to remain within a predefined range $[-c, c]$. However, this limits the flexibility of the network and, therefore, the complexity of the functions it can represent, potentially resulting in underfitting when the clipping range is too narrow. By contrast, if the range is too broad, constraining individual weights of the network may be insufficient to satisfy the 1-Lipschitz requirement globally on the critic, allowing the norm of its gradients to exceed 1.

To enforce the 1-Lipschitz constraint more flexibly, the gradient penalty (GP) term from Equation (3) is incorporated into the Wasserstein critic objective in Equation (1). This promotes more stable and responsive gradient flow and leads to the following critic objective:

$$\mathcal{L}_{\text{critic}} = -W_{\text{est}} + \lambda_{\text{gp}}\,\text{GP}. \tag{2}$$

The contribution of GP is controlled by the coefficient $\lambda_{\text{gp}}$, which regulates the strength of the constraint. Larger values of $\lambda_{\text{gp}}$ enforce the 1-Lipschitz condition more strongly by penalising large critic gradients more heavily. This limits abrupt changes in the critic output for small variations in the input and helps prevent unstable parameter updates during training. However, excessively large values of the GP's coefficient may over-constrain the network, prioritising the satisfaction of 1-Lipschitz condition over its ability to distinguish between real and generated samples. Rather than enforcing a sharp constraint on the critic weights, GP provides directional feedback based on the critic's input-output behaviour. As a result, it preserves the expressive power of the network while penalising deviations from the 1-Lipschitz condition. In this study, $\lambda_{\text{gp}} = 10$ was used throughout all WGAN-GP experiments. The gradient penalty is evaluated on interpolated samples $\hat{x}$, constructed as

$$\hat{x} = \alpha x_{\text{real}} + (1-\alpha)x_{\text{fake}}, \quad \alpha \sim \mathcal{U}(0,1),$$

which proportionally blend the real and generated samples, scaled by $\alpha$ and $1-\alpha$, respectively. The interpolation coefficient $\alpha$ is sampled independently from a uniform distribution for each sample within the batch.

Evaluating the critic on interpolated samples provides insight into its behaviour throughout the intermediate region between the real and generated probability distributions, rather than solely at the extreme endpoints. This corresponds to the region where meaningful gradients are most important for generator training, as it is where the generator can still improve. The GP for a batch of interpolated samples is then defined as

$$\text{GP} = \mathbb{E}\left[\left(\|\nabla_{\hat{x}} C(\hat{x})\|_2 - 1\right)^2\right], \tag{3}$$

where $\nabla_{\hat{x}} C(\hat{x})$ denotes the gradient of the critic output with respect to its input $\hat{x}$.

In practice, as implemented in the code, the gradient penalty penalises any deviation of the critic gradient norm from 1, whether positive or negative, on interpolated samples between the real and generated distributions. Gradient norms larger than 1 indicate excessive sensitivity of the critic to input changes, whereas norms significantly smaller than 1 correspond to regions where the critic becomes insensitive to input changes, resulting in weak or vanishing gradients for the generator. In both cases, the quality of the training signal is reduced and therefore penalised.

For this study, the WGAN-GP configuration was selected because it mitigates mode collapse and provides a more meaningful assessment of the distance between the real and generated probability distributions than the standard BCE objective combined with a sigmoid activation function. Incorporating the gradient penalty term allows the 1-Lipschitz constraint to be enforced without strictly restricting the critic weights, thereby preserving the flexibility of the network while regulating the critic behaviour throughout the intermediate region between the real and generated samples.

Both the classical WGAN-GP and the hybrid quantum-classical WGAN-GP follow the same adversarial training framework and employ the same critic architecture. The models differ only in the generator architecture, where the hybrid quantum-classical configuration replaces the classical neural network generator with a parameterised quantum circuit, described in one of the following sections.

## 2.2 Quantum Computing Fundamentals

Quantum Computing is an emerging field in computer science that integrates the principles of quantum mechanics into digital computational frameworks. Unlike classical computation, which operates on binary bits that can take values of either 0 or 1, quantum computation relies on quantum bits, or qubits.

A qubit is the fundamental unit of quantum information. While a classical bit can occupy only one of two discrete states, a qubit can be described by a quantum state represented as a linear combination of two computational basis states $|0\rangle$ and $|1\rangle$. This property is referred to as superposition and, due to the simultaneity of states, enables the encoding of significantly more complex systems than is possible with its classical counterpart. A quantum state $|\psi\rangle$ for a single qubit can be formulated as

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,.$$

where $\alpha$ and $\beta$ are complex-valued probability amplitudes corresponding to computational basis states $|0\rangle$ and $|1\rangle$, respectively. Upon measurement, only a single discrete outcome can be observed, with probabilities determined by the Born rule as to the squared moduli of the corresponding amplitudes. Therefore, for a single qubit, this yields

$$P(0) = |\alpha|^2, \qquad P(1) = |\beta|^2.$$

Since the measurement must result in one of the valid computational basis states, these probabilities satisfy the normalisation condition

$$|\alpha|^2 + |\beta|^2 = 1.$$

This condition implies that the total probability of a quantum state collapsing into any valid computational basis state is equal to 1. These properties make quantum circuits especially suitable for non-deterministic sampling problems, as the quantum state they encode can represent a probability distribution from which samples are obtained through repeated preparation and measurement of the parametrised quantum circuit. During training, the circuit parameters that determine the probability amplitudes are adjusted so that the encoded distribution approximates the target distribution.

As introduced above, superposition is an underlying property of a quantum system that allows a quantum state to exist as a combination of multiple basis states. Under ideal conditions, this superposition is maintained until measurement. The quantum state itself is fragile and must remain isolated in order to be preserved. Interaction with the environment can lead to quantum decoherence and, upon measurement, cause its collapse into a single discrete state. This transformation results from the entanglement of the qubit with its environment.

Entanglement is another fundamental property of a quantum system that can arise between multiple qubits. It introduces correlations between qubits that make their states co-dependent regardless of the distance between them. Combined with superposition, entanglement enables the representation of more complex problems through both simultaneity of states and inter-qubit correlations. In addition, measuring one qubit in an entangled system constrains the possible outcomes of measurements performed on the remaining entangled qubits.

Geometrically, a single qubit can be visualised on the Bloch sphere as a point on the surface of a unit sphere with radius $r = 1$, defined by the $X$, $Y$ and $Z$ axes. The position of the quantum state is described by two angles, $\theta$ and $\varphi$, sufficient to visualise the final state of a single qubit.

In quantum circuits, however, the state can be modified through arbitrary single-qubit rotations implemented using quantum gates, which are commonly parametrised by three angles $(\theta, \varphi, \lambda)$. These angles describe how the gate transforms the state, rather than defining the final position of the state on the Bloch sphere. Adjusting the values of these parameters therefore changes the applied rotation and consequently modifies the encoded quantum state. A visual representation of a single-qubit state on the Bloch sphere is shown in Figure 1.

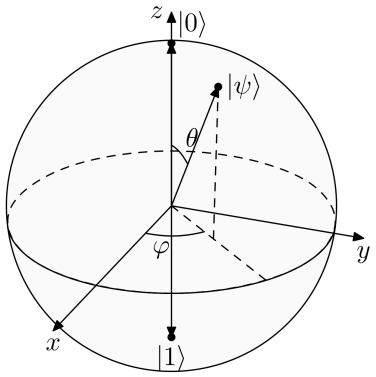


Figure 1: Bloch sphere representation of a single-qubit quantum state $|\psi\rangle$ parametrised by two angles $\theta$ and $\varphi$. Source: Wikimedia Commons [con24].

Mathematically, quantum gates are unitary transformations represented as matrices that modify quantum states. Unitary transformations have two core properties: they preserve the norm of the quantum state, ensuring that the total probability across all possible measurement outcomes remains equal to 1, and they are reversible, so that there always exists an inverse operation capable of recovering the original state prior to the transformation.
These gates can be aggregated into quantum layers that construct parameterised quantum circuits (PQCs), which are the primary trainable models in quantum computing. In circuit diagrams, each wire represents a qubit, while the gates correspond to operations applied to it. For example, the rotational $R_X$, $R_Y$, and $R_Z$ gates manipulate individual qubit states through rotations about the Bloch sphere axes, while the Controlled-NOT (CNOT) gate introduces pairwise entanglement.
Circuit depth is defined as the number of sequential quantum layers in a circuit. Deeper circuits generally increase the expressivity of the system and enable more complex representations, while also making NISQ devices more sensitive to error propagation. The capacity of a quantum circuit further depends on the interconnectivity and entanglement among qubits. Shallow circuits are characterised by a limited number of layers and, by constant depth, where the number of layers does not scale with the number of qubits. This results in a small number of processing steps that can be performed. In contrast, deep quantum circuits contain a larger number of sequential layers, and their depth may grow as the number of qubits increases, which can cause the number of processing steps to increase significantly.

# 3 Literature Review

## 3.1 Classical Generative Modelling

Generative modelling is a class of machine learning approaches developed to learn the underlying probability distribution of a dataset and generate new samples that resemble the target data. In contrast to discriminative models, which classify data into predefined categories, generative models seek to capture structures within the data itself. Such approaches have found applications across a wide range of domains, including image synthesis, text and audio generation, and scientific data modelling, such as molecular modelling for drug discovery.

Several distinct methods have been developed for generative modelling. For image generation, diffusion models currently achieve the best performance in terms of sample quality [DN21]. They learn to generate images by iteratively denoising corrupted samples [HJA20]. Apart from their capability to produce highly realistic outputs, they often require multiple sequential inference steps, resulting in significant computational costs during generation.

An alternative is provided by Generative Adversarial Networks (GANs), introduced by Goodfellow et al. [GPAM+14] and described in Section 2.1.1. GANs result in substantially lower generation costs, as images are produced through a single forward pass of the generator. They are based on an adversarial training framework, through which the generator gradually learns to produce increasingly realistic outputs. Despite their success, conventional GANs are often difficult to train and can suffer from vanishing gradients and mode collapse. To address these limitations, the Wasserstein Generative Adversarial Network (WGAN) was introduced [ACB17]. It replaces the Binary Cross Entropy objective with the Wasserstein distance and substitutes the sigmoid-activated discriminator with an unbounded critic. This configuration provides more informative gradients and mitigates several training instabilities, as discussed in detail in Section 2.1.2.

The WGAN requires the critic to satisfy a 1-Lipschitz continuity constraint, which was originally enforced through weight clipping. This approach, however, can restrict the expressive capacity of the model. To overcome this limitation, Gulrajani et al. introduced a gradient penalty (GP) term, establishing the WGAN-GP framework. This regularisation encourages the critic network to satisfy the 1-Lipschitz constraint while maintaining sufficient expressive power and improving training stability, as described in 2.1.3.

Variational Autoencoders (VAEs) are another important class of generative models [KW13]. They learn to compress data into a latent representation with an encoder and generate new samples by decoding latent vectors sampled from a predefined latent distribution. While VAEs generally offer stable training and well-structured latent spaces, their outputs often lack the sharpness and fine detail achieved by adversarial approaches. As a result, VAEs are frequently outperformed by GAN-based and diffusion-based models in image generation tasks, as they tend to capture the overall structure of the data while producing more blurry samples.

Due to the low dimensionality and limited size of the dataset, diffusion models were considered unnecessarily computationally demanding for this task, despite their state-of-the-art image generation performance. WGAN-GP was therefore selected as the classical baseline architecture, as it generates samples at substantially lower inference cost while improving upon conventional GANs through increased training stability and greater robustness to mode collapse. Alternative approaches, such as Variational Autoencoders (VAEs), were not considered further, as they typically produce less sharp outputs than adversarial models.

## 3.2 Quantum Generative Modelling

Several quantum approaches to generative modelling have been proposed, including Quantum Circuit Born Machines (QCBMs), Quantum Generative Adversarial Networks (QGANs), and Quantum Boltzmann Machines (QBMs). Parametrised quantum circuits can serve as generative models because repeated preparation and measurement of a quantum state produces samples from the probability distribution encoded by the circuit. The outcome probabilities are determined by the Born rule [LW18], introduced for the single-qubit case in Section 2.2.
For a general quantum state $|\psi\rangle$, the probability of obtaining the computational basis state $|x\rangle$ upon measurement is given by

$$P(x) = |\langle x \mid \psi \rangle|^2.$$

where $\langle x|\psi\rangle$ is the probability amplitude associated with outcome $x$. It is given by the inner product between the basis state $|x\rangle$ and the quantum state $|\psi\rangle$, and therefore quantifies the overlap between the measured basis state and the quantum state being measured.
Unlike QGAN-based models, which often combine quantum and classical components in hybrid architectures to learn a target probability distribution, QCBMs represent probability distributions directly through the measurement statistics of a quantum state and are trained by adjusting the circuit parameters to match the target distribution. Liu and Wang introduced a differentiable training framework for QCBMs based on minimising a kernel-based Maximum Mean Discrepancy (MMD) loss [LW18], which is also employed in this study.
In contrast, Quantum Boltzmann Machines (QBMs) [AAR+18] were not selected in this work because their training typically requires the preparation and sampling of quantum thermal states, which remains challenging on NISQ hardware. As a consequence, QCBMs were considered a more suitable fully quantum generator architecture for studying quantum generative modelling under the classical simulation setting used in this study.

## 3.3 Hybrid Quantum-Classical Generative Modelling

Hybrid quantum-classical generative models combine parametrised quantum circuits (PQCs) with classical elements, making them suitable for current Noisy Intermediate-Scale Quantum (NISQ) devices. In generative modelling, this often involves incorporating PQCs into existing classical frameworks, for example by replacing the generator, discriminator, or selected layers of a GAN [AOAKF+26]. Such models provide an intermediate approach between fully classical and fully quantum generators, allowing the role of quantum components to be studied while retaining established classical training procedures. For this reason, a hybrid quantum-classical WGAN-GP was included in this work as a quantum-enhanced extension of the classical baseline.

The three architectures considered in this work represent distinct generative modelling paradigms: WGAN-GP as a robust classical baseline, QCBM as a classically optimised fully quantum generator based on direct modelling of probability distribution, and the hybrid quantum-classical WGAN-GP as an adversarial model incorporating quantum components. Comparing these models enables an assessment of the potential advantages and limitations of quantum techniques for synthetic data generation under classical simulation.

# 4 Methodology

This section describes the experimental setup used to evaluate and compare three generative models on the low-dimensional Bars and Stripes dataset: a classical WGAN-GP, a hybrid quantum-classical WGAN-GP, and a Quantum Circuit Born Machine (QCBM). The section is organised as follows: Section 4.1 introduces the dataset and its preprocessing, Section 4.2 describes the architecture and training procedure of each model, Section 4.3.1 summarises the architectural, training and software implementation details, followed by Section 4.3.2, which describes the computational resources used. It is concluded with Section 4.4 that defines the evaluation metrics used to assess the generative performance.

## 4.1 Bars and Stripes Dataset

Due to the constraints imposed by the NISQ era, it is important to choose a dataset well suited for quantum computation. The choice of the Bars and Stripes dataset, restricted to low-dimensional $3 \times 3$ patterns, is motivated by the memory constraints of classical simulation of quantum systems, as discussed in Section 1.1. This restricts the number of pixels to nine while still providing a set of non-trivial patterns consisting of 14 valid configurations shown in Figure 2.

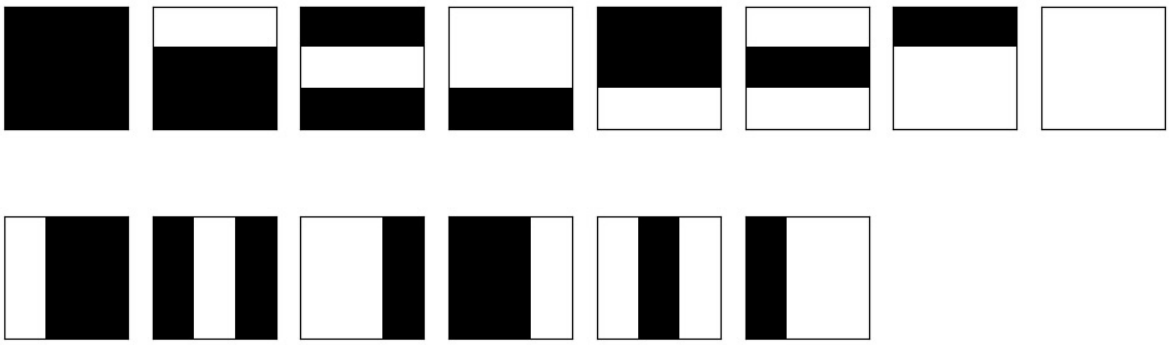

Figure 2: The 14 valid Bars and Stripes patterns on a $3 \times 3$ binary grid (black = 0, white = 1). The top row shows the 8 horizontal bar patterns and the bottom row shows the 6 vertical stripe patterns, where each row or column is uniformly black or white.

All inputs processed by both the classical and hybrid quantum-classical models are classical, represented as binary bitstrings encoding $3 \times 3$ data patterns, where black pixels correspond to zero-valued bits and white pixels to one-valued bits. The dataset requires minimal preprocessing. For the WGAN-GP models, each $3 \times 3$ binary image is flattened into a 9-dimensional vector and cast to 32-bit floating point, matching the input format expected by the generators.
Unlike the WGAN-GP models, the QCBM takes no data patterns as input. Instead, training is driven by the divergence between the generated and target probability distributions. The target distribution assigns an equal probability of $\frac{1}{14}$ to each of the 14 valid bitstrings across all $2^9 = 512$ possible states, and zero elsewhere, as shown in Figure 3, serving as the reference distribution against which the MMD loss is minimised. Increasing the dimensionality of the images while retaining

only 14 valid patterns would substantially increase the complexity of the learning task. As the dimensionality grows, the total number of possible binary image configurations grows exponentially, whereas the number of valid target patterns remains fixed. The target distribution would therefore form a sparse subset of all possible patterns, increasing the likelihood of generating samples outside its support. For this reason, such an extension is not ideal for the present study, where the objective is to first evaluate the approach under the simplest possible configuration.

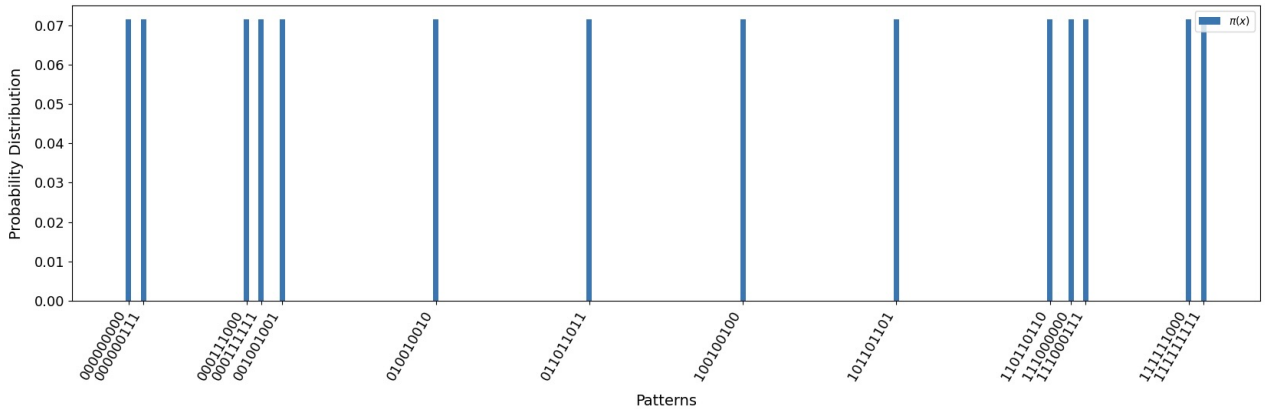


Figure 3: Target probability distribution $\pi(x)$ over the 14 valid Bars and Stripes patterns. The distribution is uniform, assigning equal probability $\pi(x) = \frac{1}{14} \approx 0.071$ to each valid pattern and $\pi(x) = 0$ to all remaining 498 states of the $2^9 = 512$ possible 9-pixel states.

No train/validation/test split is applied, as the Bars and Stripes dataset consists of only 14 fully enumerable valid patterns, and model performance is evaluated with distributional metrics rather than per-sample accuracy. As a consequence, withholding any patterns would only reduce the model's ability to learn the target distribution without providing any meaningful evaluation benefit.

## 4.2 Models

This section describes the architectures and training paradigms of the three generative models used in this work. The classical and hybrid quantum-classical WGAN-GP models follow an adversarial training paradigm, where a generator and a critic are trained in opposition to each other. The third model, the Quantum Circuit Born Machine (QCBM), represents a fundamentally different approach: rather than adversarial training, it directly optimises a quantum circuit to approximate the target probability distribution. Despite this architectural difference, both frameworks share a likelihood-free training paradigm, as their learned distributions are derived implicitly from generated samples rather than represented by a closed-form expression.
All quantum circuits presented in this study are prototypes developed with PennyLane, which provides a differentiable quantum computing framework enabling integration of parametrised quantum circuits with classical machine learning libraries such as JAX and PyTorch. Provided these prototypes demonstrate promising results, they could subsequently be deployed on real quantum devices for a more rigorous assessment of the potential quantum advantage.

### 4.2.1 WGAN-GP

#### 4.2.1.1 Architecture

The generator is a three-layer multilayer perceptron (MLP) that maps a latent noise vector $z \in \mathbb{R}^{12}$ to a 9-dimensional output corresponding to a flattened $3 \times 3$ binary pattern:

$$G : z \in \mathbb{R}^{12} \to \mathbb{R}^{64} \to \mathbb{R}^{64} \to \mathbb{R}^{9} \to [0,1]^{9}.$$

Although the dataset is limited in complexity, using a latent noise vector with higher dimensionality than the final output allows the generator to capture meaningful differences across patterns. The final latent dimension of 12 was selected through iterative tuning, as further increases did not consistently improve performance.
The MLP is characterised by fully connected layers, with ReLU activations applied across the hidden layers to enable the learning of complex non-linear functions. The final output is passed through a scaled sigmoid activation, $\sigma(4x)$, which sharpens the output activations towards $\{0, 1\}$ and encourages the generator to output binary patterns consistent with the Bars and Stripes dataset.
The critic is also a three-layer MLP that maps a 9-dimensional input to a single scalar score:

$$C : [0,1]^{9} \to \mathbb{R}^{64} \to \mathbb{R}^{64} \to \mathbb{R}$$

A LeakyReLU(0.2) activation function, as opposed to a standard ReLU, is used in the hidden layers to maintain consistent gradient flow and reduce inactive neurons by allowing non-zero gradients for negative pre-activations. These weak negative responses act as suppressive signals and influence subsequent layers and parameter updates rather than being discarded.
No output activation is applied, as the Wasserstein framework requires the critic to produce unbounded real-valued scores rather than probabilities.

#### 4.2.1.2 Training Procedure

The training process is formulated as an adversarial optimisation problem between a critic $C$ and a generator $G$ within the WGAN-GP framework. As discussed in Section 2.1.1, the critic is trained using the Wasserstein objective regularised by a gradient penalty term that enforces the 1-Lipschitz constraint and stabilises critic training. The critic loss follows the objective given in (2):

$$\mathcal{L}_{\text{critic}} = -W_{\text{est}} + \lambda_{\text{gp}}\,\text{GP},$$

where $W_{\text{est}}$ is computed as in (1):

$$W_{\text{est}} = \mathbb{E}[C(x_{\text{real}})] - \mathbb{E}[C(x_{\text{generated}})].$$

Unlike conventional GAN discriminators, the critic does not perform probabilistic binary classification. Instead, it learns a continuous 1-Lipschitz-constrained scoring function, where the difference between the average scores assigned to real and generated samples estimates the Wasserstein distance. The critic's goal is to maximise this distance, therefore sharpening the distinction between target and synthetic data distributions. In contrast, the generator aims to minimise this discrepancy

by producing samples that receive increasingly higher critic scores, closer to those assigned to real data. The corresponding reduced Wasserstein generator objective is therefore defined as

$$\mathcal{L}_{\mathrm{W}} = -\mathbb{E}[C(x_{\mathrm{generated}})].$$

This form contains only the term for generated samples, since minimising this objective encourages the generator to produce samples that better match the target patterns.
In addition to the reduced Wasserstein objective, the generator is regularised by three auxiliary loss terms: structure loss, diversity loss, and uniformity loss. These encourage the generation of structurally valid Bars and Stripes patterns, promote diversity within generated batches, and enforce equal coverage of the target distribution, respectively.
The final generator objective is therefore

$$\mathcal{L}_{\mathrm{generator}} = \mathcal{L}_{\mathrm{W}} + \lambda_{\mathrm{struct}}\mathcal{L}_{\mathrm{struct}} + \lambda_{\mathrm{div}}\mathcal{L}_{\mathrm{div}} + \lambda_{\mathrm{uniform}}\mathcal{L}_{\mathrm{uniform}}.$$

where each hyperparameter $\lambda$ controls the relative contribution of its corresponding auxiliary term. During training, the critic is updated $n_{\mathrm{critic}} = 5$ times per each generator update to provide a reliable learning signal for the generator. New latent vectors $z \in \mathbb{R}^{12}$ are sampled for both critic and generator updates, producing synthetic samples $x_{\mathrm{generated}} = G(z)$ for evaluation. Both networks are optimised using the Adam optimiser with a learning rate of $10^{-4}$ and $(\beta_1, \beta_2) = (0.5, 0.9)$ over 15,000 training epochs, where $\beta_1$ and $\beta_2$ control the moving averages of the gradients and squared gradients, respectively, which are used for adaptive learning-rate scaling.
In general, the training procedure follows an adversarial WGAN-GP optimisation process between the critic and generator, regularised by multiple $\lambda$-weighted auxiliary loss components. The generator's auxiliary loss terms are defined as follows.

**Structure Loss:** A generated sample can be represented as a matrix $X \in \mathbb{R}^{3\times3}$, where

$$X = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix},$$

with $i$ indexing rows and $j$ indexing columns. The structure loss is defined as

$$\mathcal{L}_{\mathrm{structure}} = \min\left(\mathrm{Var}_{\mathrm{rows}}, \mathrm{Var}_{\mathrm{cols}}\right),$$

where

$$\mathrm{Var}_{\mathrm{rows}} = \frac{1}{3}\sum_{i=1}^{3} \mathrm{Var}\left(x_{i1}, x_{i2}, x_{i3}\right)$$

and

$$\mathrm{Var}_{\mathrm{cols}} = \frac{1}{3}\sum_{j=1}^{3} \mathrm{Var}\left(x_{1j}, x_{2j}, x_{3j}\right).$$

The term $\mathrm{Var}_{\mathrm{rows}}$ measures the average within-row variance, and thus how much the entries within each row vary. The smaller its value, the closer each row is to a uniform structure, corresponding to a horizontal stripe pattern. Similarly, $\mathrm{Var}_{\mathrm{cols}}$ measures the average within-column variance, and

thus how much the entries within each column vary. The smaller its value, the closer each column is to a uniform structure, corresponding to a vertical bar pattern. The use of the minimum operator

$$\min\left(\mathrm{Var}_{\mathrm{rows}}, \mathrm{Var}_{\mathrm{cols}}\right)$$

implies that the loss encourages *either* horizontal or vertical structure, rather than enforcing both simultaneously; it favours the lower-variance configuration. As a result, a generated sample is considered valid and achieves zero structure loss when either all rows or all columns are constant across their entries.

**Diversity Loss:** The diversity loss mitigates mode collapse by encouraging generated samples within a batch to differ from one another. It is defined as the inverse of the mean pairwise distance between generated samples:

$$\mathcal{L}_{\mathrm{diversity}} = \frac{1}{\mathrm{MeanDist} + \epsilon},$$

where $\epsilon > 0$ is a small constant ensuring numerical stability. For a batch $\mathcal{B} = \{X^{(1)}, \ldots, X^{(B)}\}$, the pairwise distance between two generated samples $X^{(i)}$ and $X^{(j)}$ is defined as the mean absolute difference across their flattened 9-dimensional representations:

$$D_{ij} = \frac{1}{9} \sum_{k=1}^{9} \left| x_k^{(i)} - x_k^{(j)} \right|,$$

where $x_k^{(i)}$ and $x_k^{(j)}$ denote the $k$-th entries of corresponding samples. The mean pairwise distance is then computed over all distinct sample pairs:

$$\mathrm{MeanDist} = \frac{1}{I} \sum_{(i,j) \in I} D_{ij}, \quad I = \{(i,j) \mid i, j \in \{1, \ldots, B\},\ i \neq j\},$$

where $I$ denotes the set of all sample pairs within the batch, excluding self-comparisons. When the generated samples are highly similar, MeanDist becomes small, causing $\mathcal{L}_{\mathrm{diversity}}$ to increase. Minimising this loss therefore promotes diversity within generated batches.

**Uniformity Loss:** The uniformity loss encourages the generator to produce all valid patterns with equal frequency, preventing a disproportionate preference towards only a subset of patterns. It is defined as as the mean of the per-target-pattern squared deviations between the empirical distribution produced by the generator and the uniform target distribution:

$$\mathcal{L}_{\mathrm{uniformity}} = \frac{1}{K} \sum_{k=1}^{K} \left( \bar{P}_k - \pi(k) \right)^2.$$

Here, $\pi(k) = \frac{1}{K}$ defines the uniform target distribution over all $K = 14$ valid target patterns, yielding $\pi(k) \approx 0.071$ for each valid pattern $k$. The term $\bar{P}_k$ represents the average probability assigned to target pattern $T^{(k)}$ across all generated samples in the batch and is defined as

$$\bar{P}_k = \frac{1}{B} \sum_{i=1}^{B} P_{ik},$$

where $B \in \mathbb{N}$ denotes the batch size. For a given generated sample $X^{(i)}$, $P_{ik}$ represents the probability of assigning $X^{(i)}$ to target pattern $T^{(k)}$. These probabilities are obtained by applying a softmax over logits $\ell_{ik}$:

$$P_{ik} = \frac{\exp(\ell_{ik})}{\sum_{k'} \exp(\ell_{ik'})}.$$

The logits are computed from the mean squared entry-wise distance $d_{ik}$ between the generated sample $X^{(i)}$ and the target pattern $T^{(k)}$, scaled by a temperature factor $\tau$:

$$\ell_{ik} = -\tau d_{ik},$$

where

$$d_{ik} = \frac{1}{9} \sum_{j=1}^{9} \left( X_j^{(i)} - T_j^{(k)} \right)^2.$$

Here, $j$ indexes the pixel values of the flattened patterns, and $(\tau > 0)$ is a temperature parameter controlling the sharpness of the assignment across target patterns. Smaller values of $\tau$ produce softer assignments by spreading probability mass across multiple patterns, while larger values produce sharper assignments by concentrating the mass around the closest target pattern. In this work, $(\tau = 12.0)$ encourages a sharp assignment to the closest matching pattern.
Minimising $\mathcal{L}_{\text{uniformity}}$ encourages the generator to distribute probability mass evenly across all target patterns. When only a small subset of target patterns is repeatedly generated, the empirical distribution deviates from the uniform target distribution, increasing the loss. In contrast, lower loss values indicate that generated samples are distributed more uniformly across all valid patterns.

### 4.2.2 Quantum WGAN-GP

The Quantum WGAN-GP extends the classical WGAN-GP by replacing the fully classical generator with a hybrid quantum-classical generator, while retaining the classical critic and the overall adversarial training framework. The quantum generator was implemented using PennyLane's `StronglyEntanglingLayers` layer structure. Each layer consists of parameterised single-qubit `Rot` gates, representing general single-qubit rotations parametrised by three angles $(\varphi, \theta, \omega)$, followed by CNOT entangling operations that introduce pairwise entanglement between qubits.

#### 4.2.2.1 Architecture & Quantum Circuit Design

The generator combines classical preprocessing layers, a parametrised quantum circuit (PQC), and classical post-processing network. The flow of information through the generator can be visualised as

$$G : z \in \mathbb{R}^{12} \to [-\pi, \pi]^6 \to \text{Quantum Circuit} \to [-1, 1]^6 \to \mathbb{R}^{64} \to \mathbb{R}^9 \to [0, 1]^9$$

where each stage represents a successive transformation from the latent noise vector to the final flattened $3 \times 3$ pattern. The classical latent noise vectors $z \in \mathbb{R}^{12}$ are first mapped to six rotation angles bounded within $(-\pi, \pi)$ using a hyperbolic tangent activation scaled by $\pi$. These angles are then encoded into the circuit through single-qubit $R_Y$ rotations. The choice of six qubits represents a compromise between the expressive power of the model and the computational cost of

the simulation. Although the final Bars and Stripes patterns contain nine pixels, the use of classical preprocessing and post-processing layers removes the need for a direct one-to-one correspondence between the number of qubits and the output dimensionality. The quantum circuit is implemented using PennyLane's `default.qubit` state-vector simulator, which is the execution backend for the quantum component of the generator and enables the exact calculation of expectation values at the end of the circuit. The six expectation values, measured from each qubit in the Pauli-$Z$ basis, form a feature vector in the range $[-1, 1]$, which is mapped to the nine-dimensional output by a classical post-processing network. As the last step, a scaled sigmoid activation function

$$\sigma(4x) = \frac{1}{1 + e^{-4x}}$$

is applied to the post-processed outputs, constraining the generated values to the interval $[0, 1]$. The scaling sharpens the sigmoid response, encouraging outputs closer to binary values while preserving differentiability during optimisation.

The trainable component of the PQC consists of a weight tensor of shape $(n_{\text{layers}} = 2, n_{\text{qubits}} = 6, 3)$, corresponding to 36 trainable parameters in total. These parameters are distributed across the `Rot` gates in two `StronglyEntanglingLayers`, each operating on six qubits. Each `Rot` gate is parametrised by three single-qubit rotation angles, $(\varphi, \theta, \omega)$, and is followed by `CNOT` gates that entangle pairs of qubits. The successive entangling operations prevent the qubit states from evolving independently and instead form a globally correlated quantum state. The use of only two variational layers was chosen to provide sufficient circuit expressivity while limiting the simulation time associated with quantum circuits of greater depth.

#### 4.2.2.2 Training Procedure

The training procedure follows the same adversarial optimisation framework described in Section 4.2.1.2. In particular, the critic is trained with the Wasserstein loss incorporating a gradient penalty, while the generator is trained with the reduced Wasserstein loss together with three auxiliary loss terms.

The main difference from the fully classical model lies in how gradients are computed for the generator. The quantum circuit is bound to the `default.qubit` state-vector simulator using a PennyLane QNode. The QNode is defined with `interface="torch"`, which enables it to accept PyTorch tensors and integrate it with PyTorch's automatic differentiation framework. In addition, `diff_method="backprop"` is used to propagate gradients through the quantum circuit via PyTorch's built-in autograd system. This allows the quantum generator to be treated as a differentiable component within the overall optimisation loop.

This gradient computation relies on the state-vector simulation, which provides access to the underlying quantum state. In contrast, execution on physical quantum hardware would not provide direct access to the full quantum state required for backpropagation. Consequently, gradient estimation would typically require hardware-compatible differentiation methods such as the parameter-shift rule.

### 4.2.3 Quantum Circuit Born Machine

The Quantum Circuit Born Machine (QCBM) is a generative model consisting of a purely quantum generator, implemented as a parametrised quantum circuit (PQC), that is trained using classical

optimisation framework. The model implicitly encodes a classical probability distribution in the measurement probabilities of a quantum state, from which samples are generated through quantum measurement. In this study, the number of qubits directly reflects the number of output pixels, such that each QCBM output directly corresponds to one of the possible $3 \times 3$ patterns.
The circuit parameters are learned through gradient-based optimisation by minimising the Maximum Mean Discrepancy (MMD) between the target and generated probability distributions. Due to the expressive nature of the parameterised quantum circuits, the QCBM model can represent a broad range of complex target probability distributions.

#### 4.2.3.1 Architecture & Quantum Circuit Design

Unlike the hybrid quantum-classical WGAN-GP architecture, the QCBM uses a fully quantum generator without classical preprocessing of input data or post-processing of generated samples. Instead of operating on individual data samples reconstructed from latent vectors, the model is trained directly on the target probability distribution.
As illustrated in Figure 4, the architecture used in this study consists of nine qubits. Each qubit corresponds to one pixel of the considered $3 \times 3$ binary pattern and is represented as a single wire in the circuit diagram. This results in a Hilbert space of $2^9 = 512$ computational basis states, where each basis state corresponds to one possible binary bitstring $x \in \{0, 1\}^9$. The parametrised quantum circuit prepares a quantum state represented as a linear combination of all possible bitstrings $x$, with corresponding probability amplitudes $\alpha_x$:

$$|\psi\rangle = \sum_x \alpha_x |x\rangle,$$

where the probability of measuring a given bitstring $x$ is determined by the Born rule and corresponds to the squared modulus of its probability amplitude:

$$P(x) = |\alpha_x|^2.$$

The circuit is composed of six trainable PennyLane `StronglyEntanglingLayers`. Each layer consists of parametrised single-qubit `Rot` gates, defined by three angles $(\varphi, \theta, \omega)$, which change the position of the qubit on the Bloch sphere. These are followed by `CNOT` gates that entangle pairs of qubits, chosen in this study to be neighbouring qubits. The rotations are the trainable components of the circuit and reshape the quantum state by changing how probability amplitudes are distributed across possible measurement outcomes. This results in a trainable weight tensor of shape $(n_{\text{layers}} = 6, n_{\text{qubits}} = 9, 3)$, corresponding to 162 trainable parameters in total. The goal of the training procedure is to find a set of weights that adjusts the probability amplitudes of the computational basis states, allowing the resulting quantum state to approximate the target probability distribution as closely as possible.
A depth of 6 layers was selected to balance expressibility and trainability. Increasing the number of layers can improve the ability of the circuit to represent complex probability distributions, but may also reduce optimisation efficiency and, on real quantum hardware, increase noise accumulation. In this study, the lower computational cost of the QCBM compared to the WGAN-GP framework, particularly in terms of simulation time, was due to its direct MMD-based training objective and allowed the construction of a circuit with 6 layers. In contrast, the hybrid quantum-classical WGAN-GP alternative was limited to 2 layers while still requiring significantly longer training time.

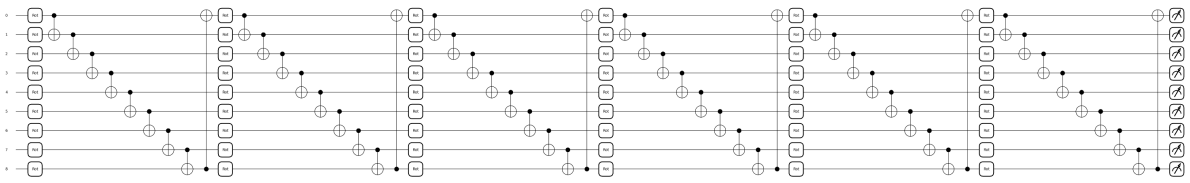

Figure 4: Circuit diagram of the 9-qubit, 6-layer QCBM. Each layer consists of trainable single-qubit `Rot` gates that control rotations of individual qubits, followed by `CNOT` gates that entangle neighbouring qubits. At the end of the circuit, all qubits are measured in the computational basis, shown by the rightmost symbols, to sample binary patterns from the learned probability distribution.

The generated samples are obtained directly from computational-basis measurements of the circuit, so no additional thresholding or classical post-processing is required.

#### 4.2.3.2 Training Procedure

The model is implemented in PennyLane using the `default.qubit` state-vector simulator, with the circuit connected to the simulator through a `qml.QNode`. The circuit returns the full output probability distribution over the computational basis states using `qml.probs()`, enabling exact evaluation under noiseless state-vector simulation. The optimisation is performed within a JAX framework, where the Maximum Mean Discrepancy (MMD) loss is computed at each step between the generated distribution $P_{\text{model}}$ and the fixed Bars and Stripes target distribution $P_{\text{data}}$:

$$\mathcal{L}_{\text{MMD}} = (P_{\text{model}} - P_{\text{data}})^{\top} K (P_{\text{model}} - P_{\text{data}}) .$$

Here, $K$ is an averaged Gaussian kernel matrix constructed using bandwidths $\{0.25, 0.5, \text{and } 1.0\}$. The use of multiple bandwidths allows the loss to capture differences between the distributions at both local and broader scales. The circuit parameters, corresponding to the weights of the trainable rotation gates, are then updated through gradient-based optimisation using the Adam optimiser, as in the WGAN-GP-based models.
The number of optimisation iterations is smaller than the 15,000 epochs used for the WGAN-GP-based models because the QCBM is not trained through an adversarial min–max procedure with alternating generator and critic updates. Instead, it directly optimises a distribution-matching objective between the generated and fixed target distributions, for which $1,500$ optimisation steps were sufficient. The learning rate of 0.1 is also substantially larger than the $10^{-4}$ used in the WGAN-GP-based models, reflecting the different optimisation dynamics of the direct MMD-based training procedure.
Each measurement of the trained QCBM in the computational basis, implemented with `qml.sample()`, produces a binary bitstring sampled according to the probabilities determined by the Born rule, as described in Section 4.2.3.1. Aggregating the outcomes obtained during the sampling procedure forms an empirical distribution that approximates the distribution encoded in the QCBM.

## 4.3 Hyperparameters & Implementation

### 4.3.1 Model Architectures and Training Configurations

| Configuration | WGAN-GP | Quantum WGAN-GP | QCBM |
|---|---|---|---|
| **Framework** | | | |
| Framework | PyTorch | PyTorch + PennyLane | JAX + PennyLane + Optax |
| Optimiser | Adam (PyTorch) | Adam (PyTorch) | Adam (Optax) |
| Precision | 32-bit | 32-bit | 32-bit |
| **Training** | | | |
| Iterations | 15,000 | 15,000 | 1,500 |
| Learning rate (G) | $10^{-4}$ | $10^{-4}$ | 0.1 |
| Learning rate (C) | $10^{-4}$ | $10^{-4}$ | — |
| Critic steps ($n_{\text{critic}}$) | 5 | 5 | — |
| Batch size (critic) | 64 | 64 | — |
| Batch size (generator) | 128 | 128 | — |
| Adam $\beta_1$ | 0.5 | 0.5 | 0.9 |
| Adam $\beta_2$ | 0.9 | 0.9 | 0.999 |
| **Loss Weights** | | | |
| $\lambda_{\text{gp}}$ | 10.0 | 10.0 | — |
| $\lambda_{\text{struct}}$ | 2.0 | 2.0 | — |
| $\lambda_{\text{div}}$ | 0.5 | 0.5 | — |
| $\lambda_{\text{uniform}}$ | 4.0 | 4.0 | — |
| **Kernel** | | | |
| MMD bandwidths | — | — | $\{0.25, 0.5, 1.0\}$ |
| **Quantum Circuit** | | | |
| Qubits | — | 6 | 9 |
| Layers | — | 2 | 6 |
| Ansatz | — | StronglyEntangling | StronglyEntangling |
| Differentiation method | — | PyTorch backprop | JAX autodiff |
| Output | — | $(\langle Z_1\rangle, \ldots, \langle Z_6\rangle) \in [-1,1]^6$ expectation-value vector | $\mathbf{b} \in \{0,1\}^9$ |
| **Architecture** | | | |
| Latent dim $z$ | 12 | 12 | — |
| Generator hidden | 64 | 64 | — |
| Critic hidden | 64 | 64 | — |

Table 1: Model architectures and training configurations for the classical WGAN-GP, hybrid quantum-classical WGAN-GP, and QCBM, as used in all experiments.

### 4.3.2 Computational Resources

All experiments were executed on the ALICE High Performance Computing (HPC) cluster hosted by the Leiden Institute of Advanced Computer Science (LIACS). The classical WGAN-GP and hybrid quantum-classical WGAN-GP models were trained on the `cpu-zen4` partition with 12 GB of RAM allocated per job. The QCBM experiments were executed on the `cpu-short` partition with the same memory allocation. To assess robustness with respect to parameter initialisation and stochastic optimisation effects, each model was trained independently using ten random seeds: 81, 49, 77, 66, 54, 30, 111, 44, 12, and 21.

## 4.4 Evaluation Metrics

To evaluate the performance of the generative models, three complementary metrics are considered: *support validity*, *support coverage*, and *Total Variation (TV) distance*. Together, these metrics assess the proportion of generated samples that lie within the valid support, the extent to which all valid patterns are covered, and how closely the generated distribution approximates the target distribution.

Evaluating generative models solely in terms of sample validity is insufficient, since a model may generate valid samples while still failing to reproduce the diversity or probability structure of the target distribution. Support coverage therefore measures whether all valid patterns are represented in the generated samples, while TV distance quantifies the discrepancy between the generated and target probability distributions through a bounded and interpretable measure of distributional similarity.

Evaluation is based on model-agnostic and interpretable metrics, since the considered models (WGAN-GP, hybrid quantum-classical WGAN-GP, and QCBM) are trained under different learning objectives, including Wasserstein-based and kernel-based approaches. For each model, the evaluation metrics are estimated from 5,000 generated samples. The assessment is likelihood-free, meaning that the generated and target distributions are compared without requiring explicit likelihood calculations.

All metrics are computed post-training for each model. For the classical and hybrid quantum-classical WGAN-GP models, samples are generated from 12-dimensional latent vectors $z \in \mathbb{R}^{12}$ and binarised element-wise using a threshold of 0.5. Although the QCBM encodes an underlying probability distribution in the trained quantum circuit, its evaluation in this work is likewise performed sample-wise to ensure consistency across models.

### 4.4.1 Support Validity

Support validity measures the proportion of generated samples that correspond to one of the valid Bars and Stripes patterns. Any sample outside the set of 14 valid patterns, represented by $\mathcal{X}_{\text{valid}}$, is considered structurally invalid.

**Validity ratio:**

$$\text{validity ratio} = \frac{N_{\text{valid}}}{N_{\text{total}}}$$

where $N_{\text{valid}}$ is the number of generated samples that belong to $\mathcal{X}_{\text{valid}}$, and $N_{\text{total}}$ is the total number of generated samples. A higher validity ratio indicates better alignment with the target support.

### 4.4.2 Support Coverage

Support coverage measures the extent to which the target support is represented in the generated samples. It quantifies whether the model captures all valid patterns in the distribution or exhibits mode collapse by failing to generate certain patterns.

**Covered patterns:**

$$\mathcal{X}_{\text{covered}} = \{x \in \mathcal{X}_{\text{valid}} \mid n(x) > 0\}$$

where $n(x)$ denotes the number of occurrences of pattern $x$ in the generated samples. $\mathcal{X}_{\text{covered}}$ therefore represents the subset of valid patterns that appear at least once in the generated dataset.

**Coverage ratio:**

$$\text{coverage ratio} = \frac{|\mathcal{X}_{\text{covered}}|}{|\mathcal{X}_{\text{valid}}|}$$

This metric measures the fraction of distinct valid patterns recovered by the model and therefore reflects its exploration of the target distribution's support. A higher coverage ratio indicates reduced mode collapse, where mode collapse is understood as the generation of only a subset of valid patterns.

### 4.4.3 Total Variation Distance

To quantify global distributional mismatch over the 14 valid patterns, the Total Variation (TV) distance is computed between the empirical distribution of generated samples, $P_{\text{model}}$, and the target data distribution, $P_{\text{data}}$, as follows:

**Total Variation:**

$$\text{TV}(P_{\text{model}}, P_{\text{data}}) = \frac{1}{2} \sum_{x \in \mathcal{X}_{\text{valid}}} |P_{\text{model}}(x) - P_{\text{data}}(x)| \,.$$

It is bounded in $[0, 1]$, where a value of 0 indicates identical distributions and a value of 1 indicates complete misalignment. In the context of a small discrete support, such as the 14 valid patterns considered here, TV distance is particularly well-suited because it provides an interpretable and comparable measure of distributional similarity. It can be computed directly from empirical samples without requiring an explicit likelihood function.
Unlike KL divergence, which measures directional information loss and is unbounded, Total Variation distance is symmetric, bounded and directly measures discrepancy in probability mass over the target support. It corresponds to the amount of probability mass that must be redistributed to transform one distribution into the other. This makes TV distance more interpretable in sparse-support settings, where some valid patterns may not be generated at all, especially since KL divergence can become arbitrarily large when the model assigns near-zero probability to regions with non-zero probability under the target distribution. These properties make TV distance particularly suitable for this study, where a consistent, comparable and model-agnostic metric is required to capture distributional differences across models.

# 5 Results

This section presents the results obtained for the three generative models investigated in this study: the classical Wasserstein Generative Adversarial Network with Gradient Penalty (WGAN-GP), the hybrid quantum-classical WGAN-GP, and the Quantum Circuit Born Machine (QCBM). The models are evaluated according to the *validity ratio*, *coverage ratio*, and *Total Variation (TV) distance* introduced in Section 4.4. These metrics quantify the proportion of valid Bars and Stripes patterns generated, the extent to which the target support is covered, and the similarity between the generated and target probability distributions.
To account for stochastic effects arising from parameter initialisation and sampling during training, each model was trained independently using ten random seeds, as specified in Section 4.3.2. Table 2 reports the mean and standard deviation of the evaluation metrics across these runs.

| **Model** | **Generative Parameters** | **Validity** | **Coverage** | **TV Distance** |
|---|---|---|---|---|
| Classical WGAN-GP | $5,577$ | $0.801 \pm 0.050$ | $0.943 \pm 0.066$ | $0.252 \pm 0.044$ |
| Hybrid WGAN-GP | 36 | $0.727 \pm 0.076$ | $0.893 \pm 0.118$ | $0.298 \pm 0.075$ |
| QCBM | 162 | $\mathbf{0.927 \pm 0.052}$ | $\mathbf{1.000 \pm 0.000}$ | $\mathbf{0.073 \pm 0.052}$ |

Table 2: Performance comparison of the considered generative models on the 3×3 Bars and Stripes dataset, including the total number of generative parameters for each model. Values are reported as mean ± standard deviation across 10 independent training seeds. Higher validity and coverage ratios indicate better performance, while lower TV distance indicates closer alignment between the generated and target distributions.

The results indicate consistent differences in performance among the evaluated models. The QCBM achieved the highest validity ratio, with an average value of $0.927 \pm 0.052$, indicating that the vast majority of generated samples corresponded to valid Bars and Stripes patterns. In comparison, the classical WGAN-GP achieved a validity ratio of $0.801 \pm 0.050$, while the hybrid quantum-classical WGAN-GP achieved the lowest validity ratio of $0.727 \pm 0.076$.
A similar trend can be observed for the coverage ratio. The QCBM achieved perfect coverage across all training runs, attaining a score of $1.000 \pm 0.000$. This indicates that all valid Bars and Stripes patterns were generated at least once during evaluation for every random seed considered. The classical WGAN-GP achieved a coverage ratio of $0.943 \pm 0.066$, whereas the hybrid quantum-classical WGAN-GP achieved a lower average coverage of $0.893 \pm 0.118$.
The strongest distinction between the models is observed in terms of Total Variation (TV) distance. The QCBM achieved the lowest distance from the target distribution, with a score of $0.073 \pm 0.052$. The classical WGAN-GP obtained a substantially larger distance of $0.252 \pm 0.044$, while the hybrid quantum-classical WGAN-GP recorded the highest TV distance of $0.298 \pm 0.075$. These results indicate that the probability distribution learned by the QCBM approximates the target Bars and Stripes distribution considerably more closely than either adversarial approach.
Overall, the QCBM achieved the best performance across all three evaluation metrics, providing the most accurate approximation of the target distribution for the considered $3 \times 3$ Bars and Stripes dataset, while the classical WGAN-GP consistently outperformed its hybrid quantum-classical counterpart.

# 6 Discussion

As results presented in Section 5 demonstrated, the QCBM achieved the strongest performance, followed by the classical WGAN-GP and the hybrid quantum-classical WGAN-GP. In particular, the QCBM recorded the highest validity ratio, complete support coverage, and the lowest Total Variation (TV) distance, indicating the most accurate approximation of the target Bars and Stripes distribution.

One possible explanation for the strong performance of the QCBM lies in its training objective. Unlike WGAN-GP models, which rely on adversarial optimisation between a generator and a critic, the QCBM directly minimises the Maximum Mean Discrepancy (MMD) loss between the generated and target probability distributions. In contrast, the WGAN-GP generators are trained indirectly by maximising critic scores, which may contribute to the observed differences in performance. Since the Bars and Stripes dataset defines a target distribution over a small, fully enumerable set of only 14 valid patterns, this direct distribution-matching objective is particularly suitable, allowing the QCBM to approximate the target distribution without the instability associated with adversarial training.

The classical WGAN-GP consistently outperformed the hybrid quantum-classical WGAN-GP across all evaluation metrics. Within the scope of this experiment, incorporating a parametrised quantum circuit into the generator did not provide a measurable advantage for the considered task. Instead, it increased training complexity without improving performance on the small-scale discrete generative task. Although the hybrid model was able to generate valid Bars and Stripes patterns, it achieved the lowest validity and coverage ratios and the highest Total Variation distance. Furthermore, the largest standard deviations observed for the hybrid model indicate the greatest sensitivity to random initialisation and sampling during training. However, the hybrid quantum-classical model used the fewest generative parameters, with only 36 compared to 5,577 in the classical WGAN-GP and 162 in the QCBM. This suggests that the reduction in parameter count did not translate into better performance. At the same time, the largest number of parameters in the classical WGAN-GP did not lead to the best performance either, despite its increased expressive capacity. This indicates that the training objective and model structure were more influential than parameter count alone.

Importantly, the observed results should not be interpreted as evidence that quantum generative models are inherently inferior or superior to classical approaches. The hybrid quantum-classical WGAN-GP and the QCBM achieved contrasting results, indicating that the effectiveness of quantum computing in generative modelling depends strongly on the selected ansatz and learning paradigm. For the considered dataset, modelling the probability distribution directly within a QCBM proved more effective than incorporating a quantum circuit within an adversarial WGAN-GP framework.

Several limitations of the present study should be acknowledged. First, the experiments were conducted on the $3 \times 3$ Bars and Stripes dataset, which contains only 14 valid patterns, and therefore represents a relatively simple generative modelling task. This dataset was intentionally selected to accommodate the computational constraints associated with evaluating quantum circuits using classical resources, as discussed in Section 4.1. For this reason, the observed results may not generalise to larger datasets or higher-dimensional probability distributions.

Second, all quantum models were evaluated through classical simulation rather than on physical quantum hardware. As a result, the experiments do not capture the effects of hardware noise, limited qubit interconnectivity, or decoherence present in contemporary NISQ devices. The results reflect mathematically modelled quantum effects under classical simulation rather than practical

quantum hardware implementations. As discussed in Section 1.1, these experiments should be treated as a preliminary assessment of quantum algorithmic prototypes, not as evidence of quantum advantage or as evidence of its absence. Further evaluation on real quantum hardware would be required to assess their practical robustness and competitiveness against both current and future classical approaches.

Third, the two quantum models differ in their circuit architectures. The QCBM uses a 9-qubit, 6-layer ansatz, where each qubit directly corresponds to one pixel of the $3 \times 3$ output pattern. In contrast, the hybrid quantum-classical WGAN-GP uses a smaller 6-qubit, 2-layer circuit, where the qubit outputs are measured as expectation-value feature vector and subsequently mapped to the 9-dimensional flattened Bars and Stripes output through classical post-processing. These architectural differences may have contributed to the observed performance gap, as the lower number of qubits and shallower circuit depth may have limited the expressivity of the hybrid quantum-classical generator. Therefore, its weaker performance may reflect not only the adversarial training framework, but also differences in circuit capacity or the interaction between the quantum and classical components.

Despite these limitations, the experiments provide a useful comparison of three distinct generative modelling paradigms. The results show that quantum generative models can learn non-trivial discrete probability distributions, but that their performance depends strongly on the chosen model type, training objective, and ansatz. For the considered low-dimensional Bars and Stripes task, the QCBM was the most effective approach, suggesting that modelling the probability distribution through quantum measurement statistics may be better suited to this setting than embedding a quantum circuit within an adversarial generator. However, these findings should be interpreted as preliminary and simulation-based, rather than as evidence of quantum advantage or its absence.

# 7 Conclusions and Further Research

This thesis investigated the performance of three generative modelling approaches on the low-dimensional $3 \times 3$ Bars and Stripes dataset: a classical Wasserstein Generative Adversarial Network with Gradient Penalty (WGAN-GP), a hybrid quantum-classical WGAN-GP, and a Quantum Circuit Born Machine (QCBM). The models were evaluated using *support validity*, *support coverage*, and *Total Variation (TV) distance* to assess the structural validity of the generated samples, the diversity of generated patterns within each batch, and the distributional similarity between the generated and target probability distributions.

The results provide an answer to the research question:

> *How does the performance of a classical Wasserstein Generative Adversarial Network with Gradient Penalty (WGAN-GP) compare to that of a hybrid quantum-classical WGAN-GP and a Quantum Circuit Born Machine (QCBM) in generative modelling, as evaluated using the validity ratio, coverage ratio, and Total Variation distance?*

Among the evaluated models, the QCBM achieved the strongest overall performance, with the highest validity ratio, complete support coverage, and the lowest Total Variation distance. The classical WGAN-GP consistently outperformed the hybrid quantum-classical WGAN-GP across all evaluation metrics, making the hybrid model the weakest-performing approach in this experiment.

These findings indicate that, for the considered $3 \times 3$ Bars and Stripes configuration, the QCBM approximated the target distribution more accurately than either adversarial approach. The results also show that introducing a parametrised quantum circuit into a WGAN-GP generator does not automatically improve generative performance. Instead, the effectiveness of quantum techniques depends on the training framework and the chosen ansatz. This emphasises the importance of assessing quantum algorithms on a model-by-model basis, rather than assuming performance improvements from the inclusion of quantum components alone.
Several directions for future research can be identified. First, the investigated models should be evaluated on larger datasets and more complex probability distributions to assess their scalability beyond the low-dimensional settings. Second, alternative quantum ansätze and training objectives could be explored to improve the performance of hybrid quantum-classical adversarial models. Third, experiments on Noisy Intermediate-Scale Quantum (NISQ) hardware would provide valuable insight into the practical feasibility of the investigated approaches under realistic hardware constraints, before fault-tolerant quantum computers become available. Finally, future work could examine whether the performance advantages observed for QCBMs under classical simulation persist as problem dimensionality, dataset complexity, and circuit size increase.
In conclusion, the results of this study demonstrate that quantum and hybrid quantum-classical generative models are capable of learning non-trivial probability distributions, but that the choice of architecture remains crucial. For the considered dataset, the QCBM, optimised using a classical framework, proved to be the most effective approach, positioning it as a promising candidate for future evaluation on real quantum hardware.

# Code Availability

The source code used in this thesis is publicly available at: https://github.com/gabrielaczapska/bsc-thesis-classical-vs-quantum-generative-models.git

# References


[AAR+18] Mohammad H. Amin, Evgeny Andriyash, Jason Rolfe, Bohdan Kulchytskyy, and Roger Melko. Quantum boltzmann machine. 2018.

[ACB17] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. 2017.

[AOAKF+26] Asma Al-Othni, Saif Al-Kuwari, Mohammad Mahdi Nasiri Fatmehsari, Kamila Zaman, and Ebrahim Ardeshir-Larijani. Hybrid quantum-classical generative adversarial networks with transfer learning. 2026.

[con24] Wikimedia Commons contributors. Bloch sphere representation. https://commons.wikimedia.org/wiki/File:Bloch_Sphere_representation.svg, 2024.

[DN21] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. 2021.

[FSD+26] Fan Fan, Yilei Shi, Mihai Datcu, Bertrand Le Saux, Luigi Iapichino, Francesca Bovolo, Silvia Liberata Ullo, and Xiao Xiang Zhu. Quantum circuit-based learning models: Bridging quantum computing and machine learning. 2026.

[GPAM+14] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. 2014.

[HBE+25] Hsin-Yuan Huang, Michael Broughton, Norhan Eassa, Hartmut Neven, Ryan Babbush, and Jarrod R. McClean. Generative quantum advantage for classical and quantum problems. 2025.

[HJA20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. 2020.

[KW13] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. 2013.

[LW18] Jin-Guo Liu and Lei Wang. Differentiable learning of quantum circuit born machines. 2018.

[OL25] Antonio D. Corcoles Constantin Dalyac Jay M. Gambetta Loic Henriet Ali Javadi-Abhari Abhinav Kandala Antonio Mezzacapo Christopher Porter Sarah Sheldon John Watrous Christa Zoufal Alexandre Dauphin Borja Peropadre Olivia Lanes, Mourad Beji. A framework for quantum advantage. 2025.

[Pre18] John Preskill. Quantum computing in the nisq era and beyond. 2018.