Master's Thesis

Explainable Quantum Machine Learning for cloud cover parametrization

Erklärbares Quantum Machine Learning für die Parametrisierung der Bewölkung

Department of Physics
Ludwig-Maximilians-Universität München

Valentina Sarandrea

Munich, April 30th, 2025

Submitted in partial fulfillment of the requirements for the degree of M. Sc.
Supervised by Dr. habil. Jeanette Miriam Lorenz

Abstract

Cloud cover parameterizations play an important role in climate models, influencing radiative transfer, atmospheric dynamics, and hydrological cycles. However, their correct formulation remains a major challenge in climate modeling, as subgrid-scale cloud processes must be approximated.
Classical and quantum Neural Networks have been proposed as potential tools for improving these parameterizations. While previous studies suggest that classical neural networks can produce physically meaningful results, it remains unclear whether quantum neural networks (QNNs) exhibit similar capabilities or rely on spurious correlations.
This study begins by comparing a classical neural network and a quantum neural network to assess whether they capture comparable physical dependencies when predicting cloud cover, before exploring alternative architectures for each approach. Using explainable AI (XAI) techniques, specifically SHapley Additive exPlanations (SHAP), the learned feature dependencies in both types of models are analyzed. This approach enables us to evaluate not only predictive performance but also the extent to which each model captures the underlying physics of cloud cover.
Our results show that both classical and quantum models exhibit similar learning patterns, extracting comparable relationships from the data. While the QNN does not outperform the classical network, it achieves comparable results, suggesting that quantum machine learning (QML) could be a viable approach in this domain. These findings contribute to the ongoing exploration of QML in climate science and highlight the potential of quantum methods for atmospheric modeling. More broadly, this study supports the integration of machine learning into climate science while ensuring physical consistency and interpretability.

KeyWords : Quantum Machine Learning, Explainable AI, Cloud Cover Parameterization, Neural Networks, Climate Modeling

Zusammenfassung

Wolkenbedeckungs-Parametrisierungen spielen eine entscheidende Rolle in Klimamodellen, da sie den Strahlungstransfer, die atmosphärische Dynamik und den Wasserkreislauf beeinflussen. Ihre korrekte Formulierung stellt jedoch weiterhin eine große Herausforderung dar, da Prozesse im Subgittermaßstab angenähert werden müssen.
Klassische und Quanten-Neuronale Netzwerke wurden als potenzielle Werkzeuge zur Verbesserung dieser Parametrisierungen vorgeschlagen. Während frühere Studien gezeigt haben, dass klassische neuronale Netzwerke physikalisch sinnvolle Ergebnisse liefern können, ist unklar, ob Quanten-Neuronale Netzwerke (QNNs) ähnliche Fähigkeiten aufweisen oder sich auf scheinbare Korrelationen stützen.
In dieser Arbeit werden beide Architekturen direkt verglichen, um zu beurteilen, ob sie vergleichbare physikalische Abhängigkeiten bei der Vorhersage der Wolkenbedeckung erfassen.
Mittels erklärbarer KI (XAI), insbesondere der SHapley Additive exPlanations (SHAP), werden die gelernten Abhängigkeiten der Modelle analysiert. Dieser Ansatz ermöglicht eine Bewertung nicht nur der Vorhersageleistung, sondern auch des physikalischen Verständnisses der zugrunde liegenden Prozesse.
Die Ergebnisse zeigen, dass sowohl klassische als auch Quantenmodelle ähnliche Lernmuster aufweisen und vergleichbare Beziehungen aus den Daten extrahieren. Obwohl das QNN der klassischen Architektur nicht überlegen ist, liefert es vergleichbare Resultate, was darauf hindeutet, dass Quantum Machine Learning (QML) ein vielversprechender Ansatz in diesem Bereich sein könnte. Diese Arbeit leistet einen Beitrag zur Erforschung von QML in der Klimawissenschaft und unterstreicht das Potenzial quantenbasierter Methoden für die Atmosphärenmodellierung. Insgesamt wird die Integration von maschinellem Lernen in die Klimaforschung unterstützt, wobei physikalische Konsistenz und Interpretierbarkeit gewahrt bleiben.

Schlüsselwörter: Quantum Machine Learning, erklärbare KI, Wolkenbedeckung, Neuronale Netzwerke, Klimamodellierung

Contents

1 Introduction

1.1 Motivation

Climate change is one of the most critical challenges of our time, shaping ecosystems, economies, and societies on a global scale [1]. The increasing concern over its impacts stems from the observed shifts in key climate variables, such as global temperature and precipitation patterns, influenced by both natural variability and human activities [2]. To understand and predict these changes, climate models play a fundamental role in studying large-scale climate variations and projecting future scenarios.

Among the various types of climate models, Earth System Models (ESMs) are the most comprehensive, as they integrate atmospheric, oceanic, land, and ice processes [3]. However, despite their sophistication, climate models show consistent bias [4], and their limitations must be addressed to improve predictive accuracy.

One of the key challenges lies in model resolution: modern climate models typically operate at horizontal resolutions between 50 and 150 km [5]. Achieving higher resolutions remains computationally prohibitive due to the computational demands of numerically solving the underlying partial differential equations (PDEs) over global scales and extended time periods, especially for ensemble runs.

Coarse resolutions result in a loss of precision because numerical solutions average physical properties over extensive areas, omitting small-scale processes that significantly influence the system's behavior. These unresolved processes include radiation, vertical diffusion, land atmosphere interactions, gravity wave drag, convection, and cloud microphysics [6]. Since these processes play a crucial role in climate dynamics, their effects must be incorporated into the model.

This necessity leads to the introduction of parameterizations, submodels that approximate the impact of unresolved processes [7, 8]. Instead of solving the governing PDEs for each microscopic detail, parameterizations estimate the average influence of these processes on the model's larger-scale variables. Parameterizations are formulated in various ways and introduce biases not only due to the selection of empirical parameters, which are often not directly measurable, but also because they are approximations that may be subject to systematic errors that cannot be eliminated, even with careful parameter selection [9]. Furthermore, different models use different parameterizations, leading to discrepancies in climate projections among various models [10].

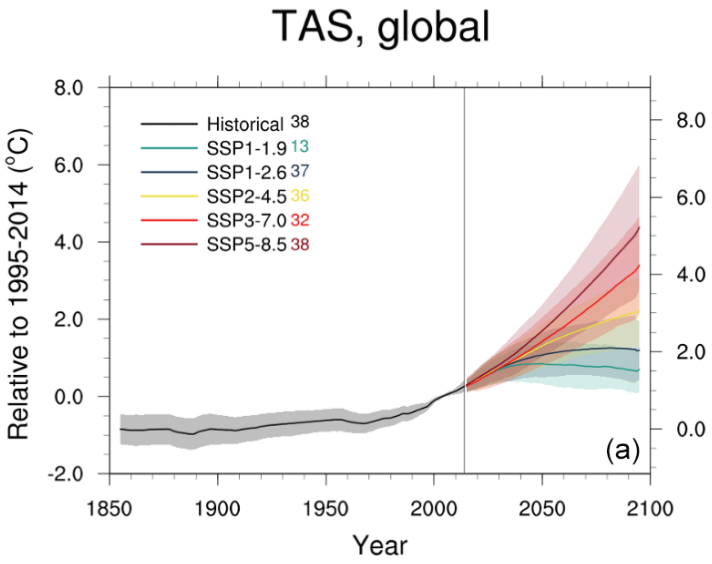

Figure 1: Global mean surface air temperature (TAS) projections from 1850 to 2100, relative to the 1995-2014 baseline. The black line represents historical observations, while the colored lines indicate future climate projections based on different greenhouse gas emission scenarios from the Scenario Model Intercomparison Project (ScenarioMIP) of CMIP6 [11]. Shaded areas represent the uncertainty range across climate models. From [10].

Clouds are fundamental components of climate models as they significantly impact Earth energy balance, regulating both incoming solar radiation and outgoing infrared radiation [12]. However, accurately representing cloud cover in models remains a major challenge, as cloud formation depends on small-scale processes that cannot be directly resolved at standard climate model resolutions [13]. To account for these unresolved dynamics, parameterizations are used to approximate their effects. Yet, even at higher resolutions, these schemes rely on statistical and empirical relationships rather than a direct physical representation of cloud processes [14]. This underscores the inherent limitations of current modeling approaches and highlights the need for improved parameterization techniques. One promising alternative is the integration of machine learning methods, which offer new possibilities for enhancing parameterization accuracy [15].

1.2 Machine Learning for Cloud Cover Parametrization

Machine learning (ML) has gained widespread recognition for its ability to tackle complex problems thanks to its highly flexible architectures. Given its success in various scientific domains, researchers have explored its potential for climate modeling [16].

In recent years, the increasing availability of high-resolution observational data and climate simulations has opened new opportunities for improving traditional parameterizations. ML approaches could potentially overcome many traditional limitations by leveraging large datasets to learn more accurate and flexible representations of physical processes [17]. Data-driven techniques enable models to learn directly from data without relying on predefined

empirical formulas. Several uses of ML for parameterizations have been proposed in the literature, with prominent applications to radiation [18–21] convection [22–27] and cloud cover [28–30]. One of the strategies in employing ML for cloud cover parameterizations is to develop data-driven schemes trained on coarse-grained data from high-resolution climate simulations, where convection and clouds are explicitly resolved.
Although ML offers a range of enhancements, it also brings forth several challenges, including the necessity for trainable models that can accommodate diverse physical scenarios, the requirement for substantial amounts of training data, and the capability to generalize across previously unseen climate regimes [17].

1.3 Quantum Machine Learning for Climate Modeling

Quantum Machine Learning (QML) is a broad field that lies at the intersection of quantum computing and machine learning. This interdisciplinary domain leverages the unique properties of quantum systems to enhance machine learning techniques, promising to solve complex problems more efficiently than classical methods [31]. Within this domain, one class of approaches involves Variational Quantum Algorithms (VQAs), which are hybrid quantum-classical algorithms. Among these, quantum neural networks (QNNs) have been explored for tasks such as classification and regression and can be trained on both quantum and classical data.
QNNs represent an alternative modeling approach compared to classical neural networks (NNs). Theoretical studies suggest that they may offer advantages in terms of generalization, expressivity, and trainability under certain conditions [32].
The number of QML applications to classical problems and datasets is rapidly growing, reflecting an increasing interest in extending the scope of quantum machine learning beyond purely quantum-related problems [33]. The use of QML is gaining traction also in weather and climate science, driven by the search for new modeling approaches [34–36].
The application of QML to classical datasets is still emerging, largely influenced by the current limitations of quantum hardware and the challenges posed by noise in quantum computations. Existing studies provide valuable insights into the field [37, 38], but it remains uncertain how significant the advantages of QML will be over classical methods in the long term [39].
In this work, we focus on the use of QNNs for regression applied to climate data, specifically for cloud cover parameterization. The motivation for investigating QML in this context lies not only in its potential to outperform classical methods in terms of computational speed or accuracy but also in its capacity to provide alternative modeling techniques that might better capture the complex, non-linear relationships in climate systems [35].

1.4 Challenges and the Role of Explainable AI

One of the major challenges in applying machine learning (ML) and QML is understanding their decision-making processes [40, 41]. This difficulty is particularly critical in climate models, where ensuring that the predictions adhere to physical laws and accurately reflect the underlying physical processes governing sub-grid phenomena is essential. As machine learning models, particularly deep learning networks, become increasingly complex,

understanding their decision-making processes has become a critical challenge. While these models often achieve high performance, their "black-box" nature makes it difficult to understand how they reach their predictions. This opacity leads to difficulties in trusting their outputs.
To address this issue, Explainable AI (XAI) encompasses a range of methods and techniques aimed at making machine learning models more interpretable and easier to understand [41–44].

XAI has already been applied in the context of machine learning for climate models, for example, to identify the most important input features for neural networks used to describe physical processes, such as cloud cover [30] or to assess the quality of the relationships a ML model has learned [27].

In this thesis, the focus is on the use of Shapley values, a well-recognized technique in XAI, to compare classical neural networks (NNs) [45, 46] and quantum neural networks (QNNs) [44, 47] for cloud cover in climate models. By analyzing the importance assigned to input features by both types of models, we aim to gain a better understanding of their decision-making processes and evaluate whether their learned representations are consistent with physical expectations and manage to capture the underlying processess important for cloud cover formation.

1.5 Key Questions

The goal of this thesis is to gain a clearer understanding of what classical and quantum neural networks are learning about the physics behind a cloud cover parameterization. In particular, using explainable AI tools, we aim to address the following key questions:

- Which atmospheric variables play the most significant role in determining cloud cover, according to the models? Do classical and quantum approaches highlight the same key factors?
- What is the physical impact of each input variable on cloud cover predictions?
- How well do the networks capture various physical regimes?
- How stable are the interpretability results? Do the patterns observed in the networks persist across different training sessions?
- How does the learning process depend on the number of input features?

1.6 Structure of the Thesis

This study builds upon the work by Lorenzo Pastori [35], who developed the classical and quantum neural networks analyzed here as part of the DLR QCI project Klim-QML [48]. The thesis is structured as follows:

- Section 2 provides all the necessary background information and prerequisites to understand the results. Specifically:

  - Section 2.1 introduces climate models, outlining their significance in simulating and predicting climate behavior.
  - Section 2.2 shows the physical processes governing cloud cover.
  - Section 2.3 delves into parameterizations, explaining how these techniques simplify complex processes in climate models to make them computationally feasible while maintaining accuracy and gives an overview regarding cloud cover schemes.
  - Section 2.4 focuses on the ICON model.
  - Section 2.5 and Section 2.6 cover machine learning and quantum machine learning.
  - Section 2.7 explains how a ML-based parameterization is created.
  - Section 2.8 introduces explainable AI, with a particular focus on Shapley values, which are employed to interpret model predictions and understand feature importance.

- Section 3 explains the architectures used for the study and the dataset used for training and testing the neural networks.

- Section 4 presents the results of the research. The subsections cover:

  - Section 4.1 focuses on the performance validation of the models, discussing the metrics used.
  - Section 4.2 presents a feature importance analysis, which investigates the contributions of different input features to model predictions.
  - Section 4.3 compares the results with the Xu-Randall approach, highlighting similarities and differences to contextualize the findings within the broader literature.
  - Section 4.4 conducts a regime-based analysis, examining how model performance varies across different climate regimes to ensure robustness and adaptability.
  - Section 4.5 evaluates different training sessions, providing insights into the stability of models' predictions.
  - Section 4.6 analyzes a model with eight input features instead of six, exploring how increasing feature complexity affects outcomes and overall model effectiveness.

- Section 5 concludes the thesis by summarizing the main results and discussing potential future developments.

2 Theoretical background

2.1 Climate models

2.1.1 Earth climate system

Climate is commonly defined as the statistical description of atmospheric conditions over a long period of time, ranging from months to thousands or millions of years [49]. The difference between weather and climate lies in their timescales: while weather exhibits short-term variability influenced by instantaneous atmospheric dynamics, climate refers to long-term statistical properties and trends in the Earth system [1].
Despite this distinction, climate and weather are intrinsically linked: climate determines the boundary conditions within weather events unfold, while the accumulation of weather patterns over time contributes to observed climate trends.
The climate system is driven by solar radiation, which provides the primary energy input. This energy is absorbed, reflected, and redistributed by different subsystems. Understanding the climate system is essential for predicting long-term environmental shifts, as well as for assessing the impact of anthropogenic activities on global temperatures and weather patterns.

2.1.2 Components of climate system

Figure 2 represents a schematic of the components of the Earth system that govern and regulate the climate.

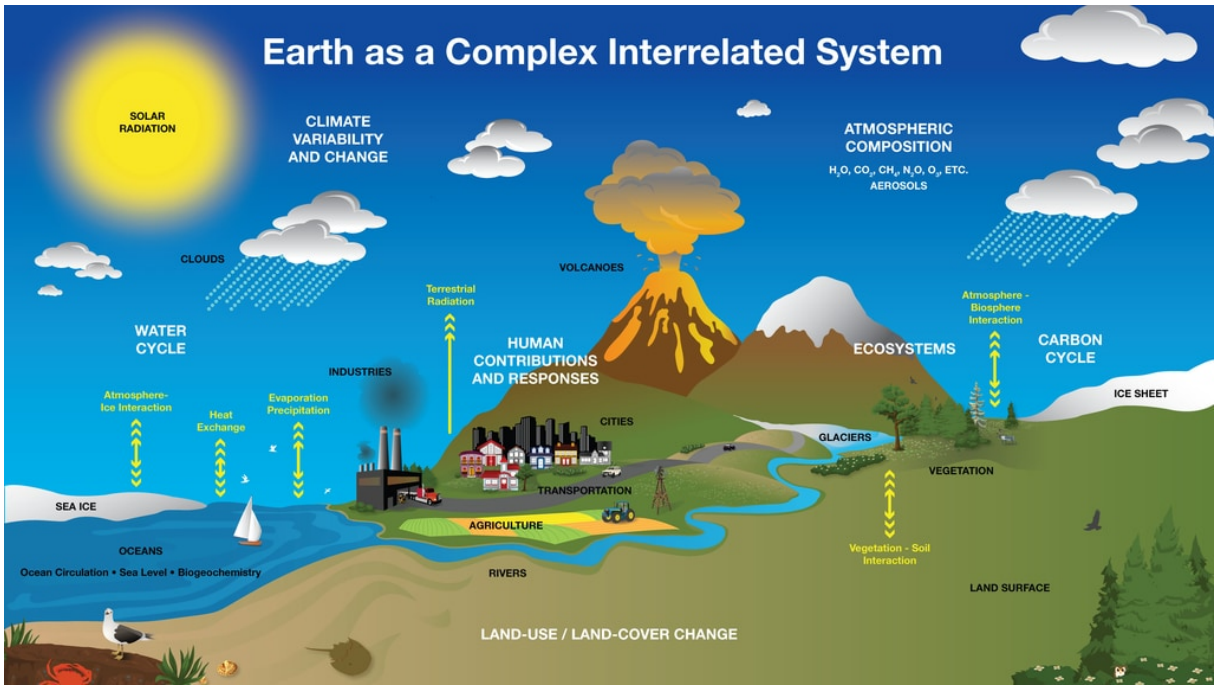

Figure 2: Diagram showing parts of the Earth system. From [50]

The Earth's climate system is determined by a complex and dynamic interplay of five major components:

- **Atmosphere**: The layer is made up of a mixture of gases, water and particles that surrounds the Earth. It plays a crucial role in regulating the planet's energy budget by absorbing, reflecting, and emitting radiation. It is divided into distinct layers: troposphere, stratosphere, mesosphere, thermosphere, and exosphere.

- **Hydrosphere** : The part of the Earth system that includes liquid ocean, inland water bodies and groundwater. It plays a central role for the water cycle, which involves evaporation, condensation, precipitation, and runoff. This cycle influences atmospheric humidity, cloud formation, and energy fluxes.

- **Cryosphere** : A subset of the hydrosphere that consist of frozen water, such as glaciers, ice sheets, sea ice, and permafrost. It is a key regulator of climate due to its high albedo, reflecting incoming solar radiation and affecting the planet's energy balance.

- **Lithosphere** : The part of the Earth system that includes the solid earth : the core, mantle, crust and soil layers. It plays a fundamental role in the carbon cycle, particularly through weathering, volcanic activity, and sedimentation processes. Land surface processes, such as soil moisture dynamics, vegetation cover, and albedo variations, affect regional climate by modulating surface energy fluxes and hydrological processes.

- **Biosphere** : A sphere that includes all of Earth's organisms, including humans, and matter that has not yet decomposed. It plays a critical role in biogeochemical cycles, particularly the carbon, nitrogen, and water cycles, which influence atmospheric composition. Terrestrial vegetation regulates surface albedo, evapotranspiration, and carbon uptake through photosynthesis, while marine ecosystems, such as phytoplankton, contribute to oceanic carbon sequestration via the biological pump. Human activities have also significantly altered biospheric processes [1].

The interactions among these components govern the planet's energy balance, water cycle, and biogeochemical processes, shaping the Earth's climate.
To represent these complex interactions and study how the climate evolves, climate models are developed as mathematical representations of the Earth's system, simulating physical, chemical, and biological processes to understand past changes and project future scenarios. Climate models vary in complexity and scope, ranging from simple energy balance models to fully coupled Earth System Models (ESMs). The choice of model depends on the research question, computational constraints, and the level of detail required.

2.1.3 Building a Climate Model: Scientific Principles and Numerical Implementation

Climate models are formulated using physical principles such as the conservation of mass, momentum, and energy. These principles lead to a set of coupled, nonlinear partial differential equations (PDEs) that describe the temporal and spatial evolution of key atmospheric and oceanic variables.
At the core of climate models are the Navier-Stokes equations, which describe the motion

of fluids and govern atmospheric and oceanic dynamics. These equations are derived from conservation laws:

- **Continuity equation (mass conservation):**

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \tag{1}$$

  where ρ is the density of air (or water in ocean models) and $\mathbf{u}$ is the velocity field.

- **Momentum equation (Navier-Stokes equation in a rotating frame):**

$$\frac{D\mathbf{u}}{Dt} = -\frac{1}{\rho}\nabla p + \mathbf{g} - 2\mathbf{\Omega} \times \mathbf{u} + \mathbf{F}_{\text{frictional}} \tag{2}$$

  This equation describes the evolution of momentum, incorporating external forces such as pressure (p) gradients, gravitational acceleration ($\mathbf{g}$), Coriolis force ($-2\mathbf{\Omega} \times \mathbf{u}$, where $\mathbf{\Omega}$ is the angular velocity of the Earth's rotation), and subgrid-scale effects modeled as a frictional term ($\mathbf{F}_{\text{frictional}}$).

- **Energy equation (first law of thermodynamics):**

$$\frac{DT}{Dt} = \frac{Q}{c_p} - \left(\frac{\partial p}{\partial t}\right)\frac{R}{c_p} \tag{3}$$

  This equation governs temperature (T) evolution, accounting for adiabatic heating (Q) from radiation, latent heat release, and convective processes. The second term represents the effect of pressure variations on temperature, where R is the specific gas constant and c_p is the specific heat capacity.

2.1.4 Numerical Implementation and Model Resolution

Since PDEs are highly nonlinear and computationally expensive to solve globally, they are discretized using numerical methods. For this, the atmpshpere, land, surface and ocean are divided into a three-dimensional grid, where each cell represents a control volume for solving the PDEs iteratively.
The resolution of a model refers to the size of the grid cells, which determines how detailed the simulation can be. Higher resolution models can resolve finer-scale physical processes but require a significantly higher computational power. For this reason, typical resolutions range from tens to hundreds of kilometers for global models [5].
Each grid cell tracks the physical state of the system, which includes variables such as temperature, pressure, humidity, and wind velocity. A climate model computes how these state variables evolve over time based on physical laws. While typical resolutions are sufficient to capture large-scale atmospheric and oceanic dynamics, many crucial small-scale processes remain unresolved, like cloud formation and microphysics, convective processes and turbulence (See Figure 3).

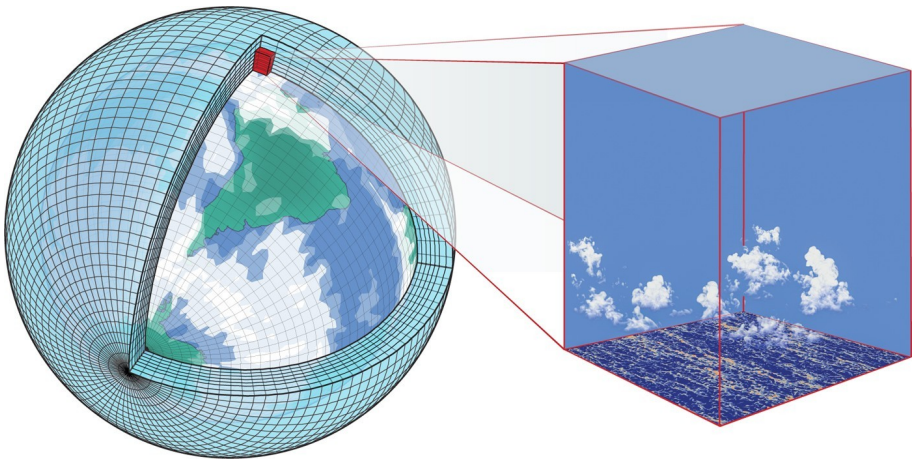

Figure 3: Illustration of the scale mismatch in climate models: Large-scale climate models operate at low resolution, while subgrid processes (like cloud formation) occur at much finer scales. These small-scale processes can not be explicitly resolved by the model [51].

Parameterizations: A Necessary Approximation To account for these unresolved processes, climate models rely on parameterizations, which are simplified representations that approximate the effect of unresolved processes on the larger-scale dynamics. Rather than solving the full set of equations governing these small-scale processes, parameterizations estimate the average effect of these processes over a given grid cell or time step. They are often based on empirical or semi-empirical relationships, derived from observations or high-resolution simulations. This approach allows the model to remain computationally feasible while still representing the impact of these unresolved processes on the larger-scale climate system. Examples of parameterizations include those for convection, cloud cover, and turbulence.

Challenges and Uncertainties in Parameterizations Parameterizations introduce uncertainties into climate models [52]. Many parameterization schemes contain free parameters, which are not known a-priori, and thus rely on empirical tuning. Also, they often are sensitive to model resolution, causing challenges such as:

- Intermodel variability: Different climate models use different parameterization schemes, contributing to uncertainties in climate projections.
- Scale dependency issues: At smaller grid spacing, the assumptions behind some physical parameterizations may not be valid, requiring updating some of the parameterization schemes.

2.2 Physical Processes Governing Cloud Cover

Cloud formation is a complex and chaotic process governed by a wide range of physical mechanisms, from microscale interactions between water vapor and aerosol particles to large-scale atmospheric dynamics.
This section provides an overview of the essential mechanisms driving cloud formation, focusing on how atmospheric variables influence their development and persistence. Rather than covering all aspects of cloud microphysics in detail, the emphasis will be on the key thermodynamic and dynamic processes necessary to understand how cloud cover is parameterized and for interpreting results from neural network-based parameterizations and assessing whether these models successfully capture the critical factors underlying cloud cover formation.

2.2.1 Cloud Formation Processes

Cloud formation is the result of complex interactions between thermodynamic, microphysical, and dynamical processes in the atmosphere. At its core, the formation of clouds requires the cooling of air masses to the point where water vapor condenses into liquid droplets or sublimates directly into ice crystals. This process is influenced by various factors, including humidity levels and the presence of aerosols acting as cloud condensation nuclei (CCN).

Adiabatic Cooling and Saturation The most common mechanism leading to cloud formation is the adiabatic cooling of rising air parcels. Adiabatic cooling requires air to rise and can happen with different lifting mechanisms, like convective lifting (leading to buoyant air parcels rising due to density differences), orographic lifting (occurs when air is forced to ascend over a topographic barrier) and convergence lifting (occurs when air masses meet and are forced upward, as in low-pressure systems and tropical cyclones).
When an air parcel rises in the atmosphere, it expands due to the decreasing pressure. This expansion leads to a decrease in temperature, as the parcel performs work on its surroundings.
As the air cools, its capacity to hold water vapor decreases. When the air parcel reaches the lifting condensation level, its temperature has dropped enough for it to be saturated, meaning that the relative humidity reaches 100%. Further cooling can lead to condensation and cloud formation. However, if the air is rapidly cooled or there are insufficient cloud condensation nuclei (CCN), the air can become supersaturated, where the relative humidity exceeds 100% without immediate condensation. This state is unstable and can lead to rapid condensation when nuclei or other disturbances trigger cloud formation.

Cloud Condensation Nuclei (CCN) and Nucleation Processes Condensation is aided by the presence of cloud condensation nuclei (CCN), which are small aerosol particles (e.g., dust, sulfate, sea salt, organic compounds) that provide a surface for water vapor to condense. The ability of an aerosol particle to act as a CCN depends on its chemical composition and size. Hygroscopic particles, such as sulfates and sea salt, are particularly effective at attracting water vapor due to their ability to absorb moisture from

the surrounding air. This property allows them to promote condensation even when the relative humidity is slightly above 100%, a condition known as low-level supersaturation.

2.2.2 Atmospheric Variables Influencing Cloud Cover

Cloud formation is influenced by a variety of atmospheric factors, which interplay in complex ways to determine the formation, persistence, and distribution of clouds. These factors include temperature, pressure, humidity, and dynamic atmospheric processes such as winds. In this section, we explore the influence of these atmospheric variables on cloud formation.

Temperature Temperature affects the capacity of air to hold moisture. Warmer air can hold more water vapor, which can be critical in determining the amount of water available for cloud formation. Conversely, in colder conditions, the air reaches saturation more easily, leading to enhanced condensation and cloud formation.

Humidity: Specific and Relative Humidity Humidity, both specific and relative, is crucial for understanding cloud formation and the persistence of clouds in the atmosphere.

- **Specific Humidity** is the mass of water vapor per unit mass of air, typically expressed in grams or kg of water vapor per kilogram of air. Specific humidity directly affects the potential for cloud formation. When air becomes saturated with water vapor and if there are surfaces for it, condensation occurs. The higher the specific humidity, the greater the potential for cloud formation.
- **Relative Humidity** is the ratio of the current amount of water vapor in the air to the amount in saturated air at a given temperature and pressure, expressed as a percentage. When relative humidity reaches 100%, the air is saturated, and condensation can occur.

Cloud Condensate Clouds can be composed of both liquid water and ice, with their physical state depending on the temperature and altitude at which they form. If there is no condensate there is no cloud.

- **Cloud water** refers to the liquid phase of water within clouds, primarily consisting of tiny water droplets suspended in the atmosphere. These droplets are formed when water vapor in the air condenses onto nuclei when the air reaches its dew point. The higher the relative humidity, the greater the potential for cloud water formation. In warm clouds, cloud water is the dominant phase.
- **Cloud ice** forms when water vapor directly transitions into ice in colder cloud environments, typically at temperatures below freezing. Ice crystals are an essential component of many types of clouds, especially in cold clouds at high altitudes. In clouds where the temperature is below the freezing point, water vapor can directly deposit onto ice nuclei, forming ice crystals. Ice can also form from cloud water droplets.

2.3 Parametrizations and Cloud Cover Schemes

Parameterizations are a tool used in climate models to represent subgrid-scale processes that cannot be explicitly resolved due to computational constraints. Rather than simulating these processes in full detail, parameterizations approximate their net effect at the grid scale of the model, ensuring consistency with the larger-scale dynamics.
Cloud cover parameterizations, in particular, estimate the fractional cloudiness within a grid cell.

2.3.1 Cloud Cover Parametrization

The simplest cloud cover parameterization considers the total cloud condensate content of a grid cell. If it exceeds a given threshold, then the grid cell is deemed fully cloudy, otherwise it is cloud-free. However, this simple approach is only reasonable in small grid cells at very high resolutions, where clouds typically fill entire grid cells. At resolutions common in Earth System Models, the fractional cloudiness needs to be estimated instead [53].

2.3.2 Sundqvist Scheme

The scheme of Sundqvist [54] explicitly expresses cloud cover as a monotonically increasing function of relative humidity (RH). This scheme assumes that clouds can only form when the grid-averaged RH exceeds a critical threshold RH_0, which depends on the ratio between surface pressure (p_s) and local pressure (p):

$$RH > RH_0 \stackrel{\text{def}}{=} RH_{0,top} + (RH_{0,surf} - RH_{0,top}) \exp\left(1 - (p_s/p)^n\right) \tag{4}$$

where $RH_{0,top}$ and $RH_{0,surf}$ represent the critical RH values at the top of the atmosphere and the surface, respectively, and n is a shape parameter controlling the vertical variation of RH_0. The ratio p_s/p represents how the critical relative humidity RH_0 varies with altitude.
When this condition is met, the cloud cover fraction C is given by:

$$C_{Sundqvist} \stackrel{\text{def}}{=} 1 - \sqrt{\frac{min\{RH, RH_{sat}\} - RH_{sat}}{RH_0 - RH_{sat}}} \tag{5}$$

where $RH_{sat} \approx 1$ represents the relative humidity in the cloudy part of the grid cell.
This scheme includes four tuning parameters that remain constant throughout a General Circulation Model (GCM) simulation:

- $RH_{0,surf}$: the critical relative humidity at the surface
- $RH_{0,top}$: the critical relative humidity in the upper atmosphere
- n : the shape factor controlling the vertical profile of RH_0
- RH_{sat} relative humidity in the cloudy portion of the grid cell (typically close to 1).

The derivation of Equations (4) and (5) relies on two assumptions:

1. The relative humidity in the cloudy portion of a grid cell remains approximately constant at RH_{sat}, while the relative humidity in the cloud-free portion is denoted by RH_{crit}. This leads to a grid-mean RH expressed as:

$$RH = CRH_{sat} + (1 - C)RH_{crit}, \tag{6}$$

   where C is the cloud cover fraction and RH_{crit} represents the relative humidity in the cloud-free portion of the grid cell. In this framework, RH_{crit} depends on factors such as temperature, land fraction, and altitude.

2. The relative humidity in the cloud-free portion of the grid cell increases linearly with the cloud fraction C.

The Sundqvist scheme provides a computationally efficient way to represent fractional cloud cover, ensuring a smooth transition between clear and cloudy conditions rather than a binary switch. However, this threshold-based approach becomes less accurate at the spatial resolutions typically used in GCMs, where clouds rarely occupy entire grid cells. Therefore, more sophisticated schemes have been developed to estimate fractional cloudiness.

2.3.3 Xu-Randall Scheme

Unlike the Sundqvist scheme, which relies solely on relative humidity, the Xu-Randall scheme [55] also incorporates cloud condensate mixing ratios. This allows it to better capture variations in cloud cover, as it ensures that grid cells remain cloud-free in the absence of condensates. The additional dependence on cloud condensate makes this scheme more physically realistic and has been shown to improve agreement with CloudSat observations [56]. In a simplified form, it can be formulated as:

$$C_{Xu-Randall} \overset{\text{def}}{=} min\{RH^{\beta}(1 - \exp(-\alpha(q_c + q_i))), 1\} \tag{7}$$

where q_c is the cloud cover mixing ratio, q_i the cloud ice mixing ratio, and $\{\alpha, \beta\}$ are two tuning parameters.
Relative humidity based cloud cover schemes generally have some notable limitations. First, the relationship between RH and cloud cover is not always well-defined observationally. For instance, Walcek [57] demonstrated that cloud cover probability can be nearly uniform even at RH values of 80%. Additionally, most cloud cover schemes rely on local thermodynamic variables, yet rapid advection (e.g., updrafts) can introduce non-local effects. To mitigate these inaccuracies, they include several tuning parameters, which are adjusted to maintain a well-balanced top-of-the-atmosphere energy budget [6].

2.4 ICON model

The ICON model (ICOsahedral Nonhydrostatic model [58]) is a flexible, scalable, high-performance modelling framework for weather, climate predictions and projections.
It is structured into different components, allowing it to simulate various aspects of the Earth system. These components include:

- **ICON-A (Atmosphere)**, the core module used for weather prediction and climate simulations.
- **ICON-O (Ocean)**, that models ocean circulation and can be coupled to ICON-A for climate simulations with coupled atmosphere and ocean dynamics.
- **ICON-L (Land)**, which represents land surface processes
- **ICON-ART (Aerosols and Chemistry)**, used for atmospheric composition modeling

ICON-ESM (Earth System Model) is the fully coupled climate system model, used for long-term climate simulations and IPCC climate projections.
Since the dataset used for training the networks in this thesis originates from ICON-A, the focus will be on this component.

2.4.1 ICON-A : Main Computational Components

The ICON atmosphere model predicts the spatio-temporal evolution of the atmospheric state in terms of the prognostic variables virtual potential temperature, 3D wind, total air density and mass fractions of atmospheric water constituents and trace gases. In its climate configuration, ICON-A employs the ECHAM physics package [6].
Mathematically, the dynamical core of ICON-A solves the fully compressible, non-hydrostatic Navier-Stokes equations on the sphere. These equations govern atmospheric fluid motion and account for a wide range of scales, from synoptic weather systems to mesoscale turbulence.
The ICON-A model consists of three main computational components [59]:

- **Dynamics** : The core of the model, responsible for solving the discrete fluid motion equations.
- **Tracer Advection** : Governs the transport of atmospheric tracers, such as humidity and cloud water.
- **Physics** : Includes parameterizations for subgrid-scale processes, such as radiation, convection, and cloud microphysics, which cannot be explicitly resolved by the dynamical core.

The following diagram summarizes the structure and workflow of the ICON model:

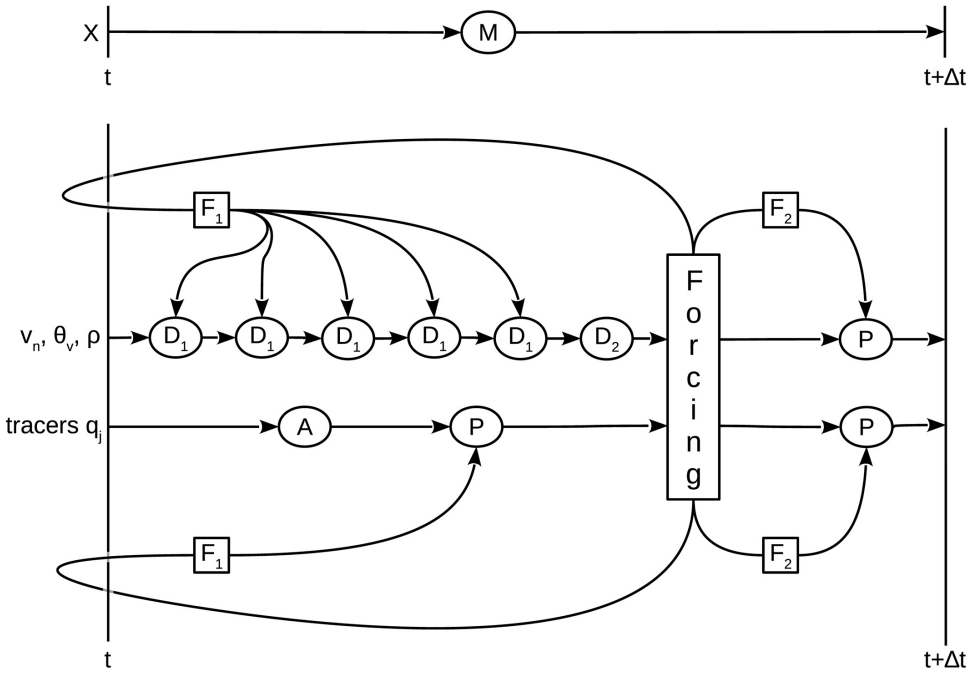

Figure 4: The model M propagates the state X from time t to the new time t+Δt (upper part). The model operator M is split in operators for dynamics D, advection A, and physics P, which yield partial updates. The dynamic variables of X (v_n, θ_v, and ρ) are processed n times by the fast dynamics operator $D1$, here shown for $n = 5$, followed by the damping/diffusion operator $D2$. The fast dynamics can be forced by the forcing from slow physics, $F1$. For efficiency reasons, a distinction is made between so-called fast physics processes, whose time scale is comparable or shorter than the model time step, and slow physics processes whose time scale is considered slow compared to the model time step. Tracer fields are first advected and then updated with the forcing from the slow physics. After dynamics and tracer advection, including the slow physics forcing $F1$, the forcing is newly computed, and the forcing owing to fast physics, $F2$, is applied to dynamics variables as well as tracer variables. Source: [6]

2.5 Machine Learning for Parametrizations

Machine learning (ML) is a widely used technology that plays a key role in data analysis, allowing classification, clustering, and pattern recognition in large datasets [31].
Its strength lies in its ability to analyze vast amounts of data and to identify hidden patterns.
There are different types of ML methods depending on the necessities, like decision trees, neural networks and support vector machines.
As in many other scientific fields [60, 61], researchers are exploring the potential of ML to enhance climate models. While ML has shown promising results, the use of ML in this field introduces several challenges, such as the need for complex yet trainable models to encompass various physical scenarios, the requirement for large amounts of training data, and the ability to generalize to unseen climate regimes [17]. For this reason, its integration into climate science is still in an on-going research field.

One application of ML in climate modelling is the development of machine learning-based parameterizations. This growing field can be broadly classified into two groups:

- The first group includes studies that use ML to emulate and accelerate existing parameterizations, such as those by [62] and [21].
- The second group includes studies that use ML to learn parameterizations directly from three-dimensional, high-resolution data. In most cases, the high-resolution data is coarse-grained to the low-resolution grid of the climate model. The first proof of concept in this area was established by [22], who trained a small NN on coarse-grained regional data. Later, various other authors adapted this approach to global models (e.g. [16, 25, 26]).

Typically, ML-based parameterizations are first developed in an offline setting, where they are trained and evaluated on pre-existing datasets before being coupled online within a full climate model. Many of the studies mentioned above, such as [23, 25, 26], focus primarily on offline training, assessing the ability of ML models to reproduce sub-grid scale processes from high-resolution simulations. However, integrating these parameterizations into online coupled simulations introduces additional challenges, such as ensuring numerical stability and physical consistency over long-term climate runs. The transition from offline to online coupling remains an active area of research, as even well-performing offline models can lead to instabilities or biases when coupled interactively within a climate model.
One ML model analyzed in this thesis is a neural network (NN) for cloud cover parameterization. In the next section, a brief description of the working principles of neural networks will be provided.

2.5.1 Neural Networks

A neural network (NN) is a computational model inspired by the human brain, designed to recognize patterns in data. It consists of layers of interconnected neurons, where each neuron performs a mathematical operation and passes its result to the next layer [63]. The goal of a neural network is to learn a function that maps input data to an output, which can be used for tasks such as classification or regression.
A feedforward neural network (FNN) is a type of NN in which data flows in one direction, from the input layer to the output layer, passing through one or more hidden layers [64]. Each neuron in a layer is connected to all neurons in the subsequent layer, and the network learns to approximate the target function by adjusting its internal parameters. The output of each neuron is computed as a weighted sum of the inputs, passed through a nonlinear activation function, which allows the network to capture complex patterns in the data:

$$f_\theta(\mathbf{x}) = \mathbf{W}^{(L)} \prod_{l=1}^{L-1} \sigma\left(\mathbf{W}^{(l)}\mathbf{x} + \mathbf{b}^{(l)}\right) + \mathbf{b}^{(L)} \tag{8}$$

Where:

- $f_\theta(\mathbf{x})$: The output of the neural network, as a function of the input $\mathbf{x}$ and the parameters $\theta = \{\mathbf{W}^{(1)}, \mathbf{b}^{(1)}, \ldots, \mathbf{W}^{(L)}, \mathbf{b}^{(L)}\}$.

- $\mathbf{x}$: The input vector to the network.
- $\mathbf{W}^{(l)}$: The weight matrix for layer l, with dimensions (n_l, n_{l-1}), where n_l is the number of neurons in layer l.
- $\mathbf{b}^{(l)}$: The bias vector for layer l.
- $\sigma(\cdot)$: The activation function.
- L: The total number of layers in the network.

The training of an FNN involves optimizing its parameters (weights and biases) by minimizing the difference between the predicted output and the actual target value. This process follows these steps:

1. **Feedforward** : The input data is passed through the network, layer by layer, to produce a prediction.
2. **Loss Computation** : A loss function measures the discrepancy between the predicted and actual values.
3. **Backpropagation** : The gradients of the loss function with respect to the weights and biases are computed.
4. **Parameter Update** : The network parameters are updated iteratively using an optimization algorithm, which adjusts the weights in the direction that minimizes the loss.

During training, the weights are updated iteratively using optimization techniques like stochastic gradient descent (SGD) [65]. Over iterations, this process helps the network learn the underlying relationships in the data.

2.6 Quantum Machine Learning

Quantum machine learning (QML) is an emerging research field that lies at the intersection of quantum computing and machine learning [31]. QML explores ways to harness quantum properties like superposition, entanglement, and quantum interference to improve machine learning models. The potential benefits include exponential speedups for certain types of optimization, more expressive models, efficient handling of complex correlations, especially in domains where classical methods struggle with high-dimensional interactions [38, 66, 67]. However, while QML presents intriguing advantages, its practical impact remains largely theoretical, as the field is still in its early stages. Recent technological advancements and promising initial results have fueled growing interest in QML [68]. Current research focuses on identifying tasks where QML can provide advantages over classical methods and on developing hybrid quantum-classical approaches that leverage the strengths of both paradigms [69].

2.6.1 Classification of Quantum Machine Learning Models

Quantum machine learning (QML) models can be categorized based on the nature of both the data and the computational components (i.e., the algorithms or devices involved). This classification helps to understand how quantum and classical elements can be combined within hybrid learning architectures [70]. As shown in Figure 5, , there are four possible scenarios depending on whether the data and the algorithm are classical (C) or quantum (Q). While only the three cases involving at least one quantum component fall under the QML umbrella, the fully classical case is included in the diagram for completeness and as a reference point.

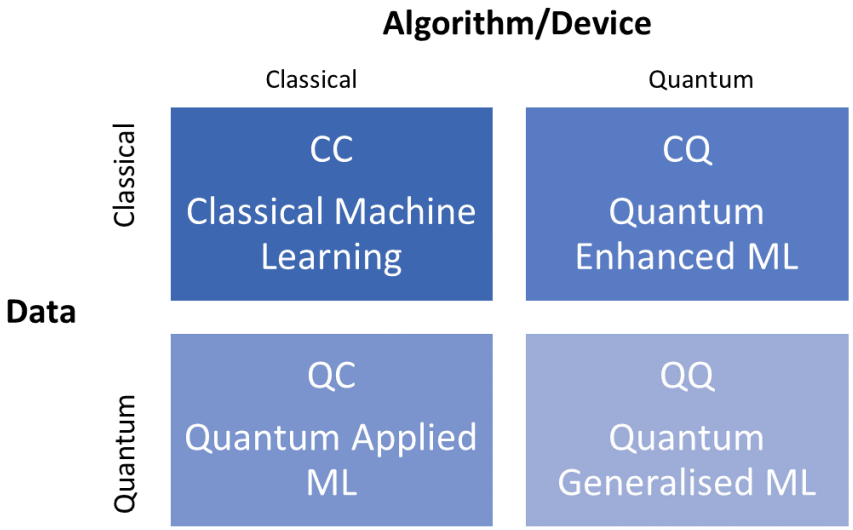

Figure 5: The QML diagram represents data and and algorithm or device, which can be classical (C) or quantum (Q) in four different scenarios. After [70]

- **Classical Data + Quantum Algorithm (C-Q)** : In this case, classical data is used as input, but the learning algorithm is quantum. Since quantum computers process information in quantum states, classical data must be encoded into quantum states using one of various encoding techniques. After encoding, quantum circuits perform computations, and the results are measured and converted back into classical outputs.
- **Quantum Data + Classical Algorithm (Q-C)** : Here, the data itself is quantum in nature, but classical algorithms are used for processing. Quantum data can come from any quantum device (e.g., a quantum simulator or a quantum sensor), typically in the form of quantum states. In this scenario, classical machine learning techniques are applied to analyze quantum datasets.
- **Quantum Data + Quantum Algorithm (Q-Q)** : This represents the most quantum-native scenario, where both data and algorithms are quantum. Quantum-

generated data is processed using quantum machine learning models that run entirely on quantum hardware.

Applications to classical problems and datasets are rapidly increasing [71], potentially broadening QML's relevance beyond purely quantum problems. This includes growing interest in applying QML to climate and weather science [34, 36, 72–74]. However, the application of QML to classical data remains challenging due to the noise and limited scalability of current noisy intermediate-scale quantum (NISQ) devices [75]. These devices, characterized by a relatively small number of qubits and high error rates, are not yet capable of executing large-scale quantum computations without significant error correction, making the long-term quantum advantage of QML still an open question [39, 76].
Nonetheless, preliminary research suggests that QML might hold significant potential for advancing climate modeling. As quantum technology progresses, it is likely to play an increasing role in climate science, offering more powerful tools to address challenges related to climate change.
This study falls into the **CQ** category, employing a hybrid quantum-classical approach, where classical climate data are processed using a QNN. In this case, classical data coming from climate models are used as input, but the learning algorithm is quantum.

2.6.2 Quantum Neural Networks

QNNs are a class of hybrid quantum-classical models inspired by classical neural networks but adapted to leverage quantum computing principles. They are a specific class of hybrid quantum-classical models that are executed on both quantum processors as well as on classical processors to perform a single task [77]. Unlike traditional deep learning architectures, QNNs use quantum circuits as computational layers.

Structure of QNNs A QNN is made up by components that loosely resemble those of classical neural networks. We will show how a QNN is made, focusing on the architecture used in this thesis:

- **Input Layer (Data Encoding)** : Since quantum computers operate on quantum states, classical data must first be mapped onto qubits. This process is known as quantum data encoding. The choice of encoding scheme impacts the model's expressiveness and efficiency [78, 79]. Common encoding techniques include:
  - Amplitude Encoding: Encodes a classical vector into the amplitudes of a quantum state.
  - Angle Encoding: Maps classical features to rotation angles of qubits in the Bloch sphere.
  - Basis Encoding: Represents data as a binary string corresponding to computational basis states.

  In this work, Angle Encoding is employed as the data encoding strategy.

- **Parameterized Quantum Circuits (PQC) as "Hidden Layers"** : Similar to the hidden layers in classical neural networks, QNNs employ a sequence of quantum gates forming a parameterized quantum circuit (PQC). These gates introduce trainable parameters and can be adjusted to optimize the network's performance. A typical PQC can be described by a unitary operator $U_\vartheta(x)$:

$$\hat{U}_\vartheta(x) = \prod_{l=1}^{L} \left(\hat{V}(\vartheta^{(l)}) \hat{S}(x) \right) \tag{9}$$

  where:

  - $\hat{S}(x)$ is the encoding layer that maps input data x to quantum states,
  - $\hat{V}(\vartheta)$ represents trainable quantum gates (variational layers) depending on parameters ϑ,

  Data re-uploading is often used to enhance expressivity, meaning that input data is re-encoded multiple times within the circuit [78].

- **Measurement** After quantum processing, the network's output is extracted by measuring an observable M

$$f_\theta(x) = \langle 0 | \, U_\theta^\dagger(x) M U_\theta(x) \, | 0 \rangle \tag{10}$$

  This collapses the quantum state into classical values. These measurements serve as the output of the QNN and can be further processed using classical techniques.

Training a QNN The training of a QNN involves optimizing the parameters θ using a hybrid quantum-classical approach. The process follows these steps:

1. **Forward Pass**: The input data is encoded into a quantum state, processed through the PQC, and measured to obtain an output.
2. **Loss Computation**: A classical loss function quantifies the difference between the predicted output and the target value.
3. **Gradient Estimation**: The gradients of the loss function with respect to the quantum circuit parameters are estimated using quantum differentiation techniques such as the parameter-shift rule.
4. **Parameter Update**: Classical optimization algorithms (e.g., gradient descent, Adam) adjust the parameters to minimize the loss.

This iterative process is repeated until convergence, allowing the QNN to approximate the underlying data distribution. In this work, QNN computations are simulated classically, meaning that the computations of the QNN, which would otherwise take place on a quantum device, are calculated numerically.

2.7 ML based parametrizations

The development of an ML-based parameterization follows a systematic process to ensure that the network learns relevant physical relationships while operating at the target resolution. For the case of cloud cover the procedure can be summarized as follows:

1. **Data Generation** : High-resolution simulations (e.g. 5km) from storm-resolving models are used as the training dataset. These simulations explicitly resolve some of the small-scale processes that are otherwise parameterized in lower-resolution models, such ad deep convection and gravity waves. Explicitly resolving these dynamical processes also improves the representation of other variables such as cloud cover.

2. **Coarse-Graining** : The high-resolution data are coarse-grained to match the resolution of the target model (e.g. 80km). This step ensures that the network is trained on data that correspond to the resolution at which it will be deployed.

3. **Training** : The dataset is used to optimize the network parameters θ so that the predicted cloud cover, $f_\theta(\overline{x})$, approximates the true coarse-grained cloud cover $clc(\overline{x})$. This optimization is performed by minimizing a suitable loss function.

4. **Implementation and Testing** : Once trained, the network is evaluated to ensure that it generalizes well to unseen data and effectively represents the subgrid-scale processes at the target resolution.

The following figure provides a schematic representation of the training process.

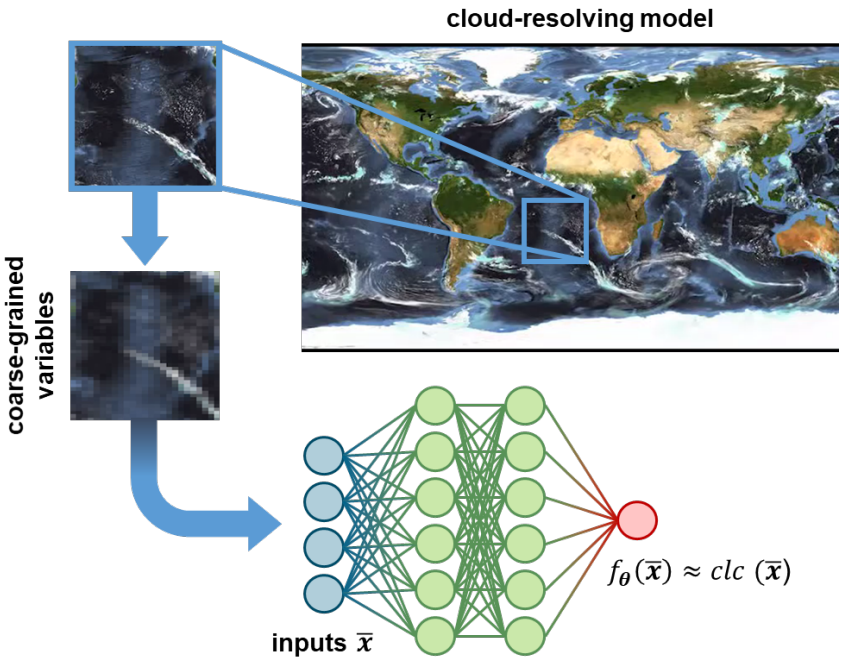

Figure 6: Schematic of the approach used to develop both classical and quantum NN-based parameterizations. High-resolution simulation data x are first coarse-grained to match the target resolution. A training dataset is thus constructed, where the coarse-grained state variables, $\overline{x}$, serve as inputs, and the corresponding coarse-grained cloud cover, $clc(\overline{x})$, acts as the output. The networks are trained by optimizing their parameters, θ, so that the predicted output, $f_\theta(\overline{x})$, closely approximates $clc(\overline{x})$. The functional form of $f_\theta(\overline{x})$ depends on whether a classical (Equation 8) or quantum (Equation 10) neural network is used. From [80]

2.8 Explainable AI methods : Shapley Values

ML models, especially complex ones such as neural networks, have demonstrated outstanding predictive performance across a variety of tasks and fields [81–83]. However, they are often regarded as "black boxes" due to their inherent lack of transparency in decision-making processes [42, 84]. This opacity is a significant concern, particularly in fields where understanding the reasoning behind a model's prediction is critical, including climate modeling.
While ML models can achieve high accuracy, this success is often paired with a substantial trade-off: the difficulty of interpreting how models make predictions (Figure 7).

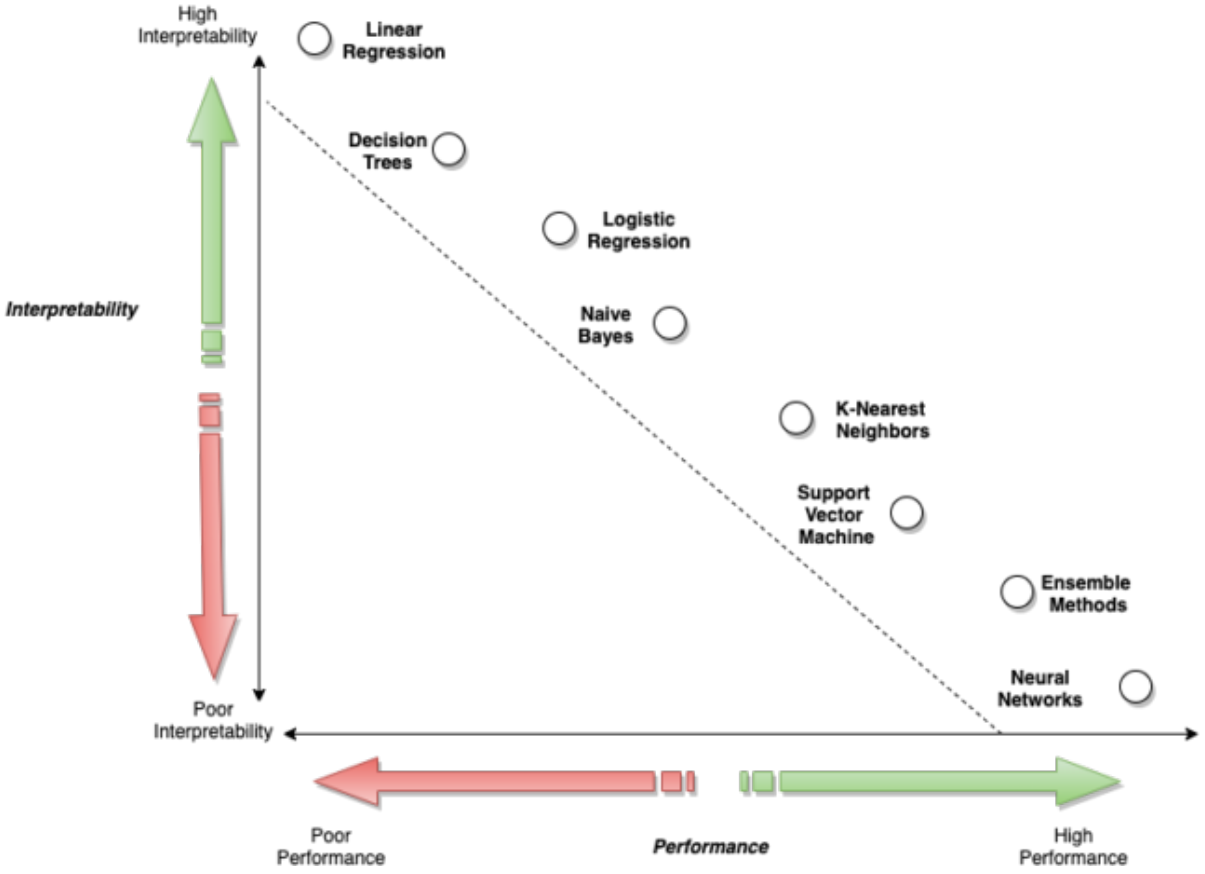

Figure 7: Trade-off between interpretability and performance in AI systems. As machine learning models improve in performance, they often become more complex and harder to interpret. Source: [85].

2.8.1 Explainable AI Categorization

Explainability refers to methods that make the behavior of ML systems or, more generally, artificial intelligence (AI) systems comprehensible for humans. The field of Explainable AI (XAI) encompasses various approaches to providing transparency in machine learning models. Realizing XAI is a highly non-trivial task with a potentially great impact on many applications and can therefore be considered as an important research field. These approaches can be categorized based on the nature of the explanation, the level of analysis,

and the compatibility with specific models. Following [42], we explore the primary categories of explainability methods:

- **Global vs. Local Explainability**:
  - **Global XAI** aims to explain the overall behavior of the model, typically by providing insights into the importance of features across the entire dataset.
  - **Local XAI**, on the other hand, focuses on explaining individual predictions. The goal here is to understand why the model made a specific decision for a given input.
- **Model-Specific vs. Model-Agnostic Explainability**:
  - **Model-specific explainability** refers to methods that are designed for particular types of models.
  - In contrast, **model-agnostic explainability** refers to techniques that can be applied to any machine learning model, regardless of its architecture. A prime example are Shapley values, which offer a fair and systematic approach for attributing the contribution of each feature to a given prediction [86].

The choice between different XAI approaches depends on the model being used and the specific requirements of the application.
This method is applicable to both classical and quantum machine learning models, and can offer both global and local explanations.

2.8.2 Shapley Values

Shapley values, a widely used method for feature attribution, were originally developed in cooperative game theory by Lloyd Shapley [87]. This approach provides a theoretically robust framework to fairly distribute the contribution of input features in a predictive model, ensuring an equitable quantification of feature importance.
The key idea is to treat the prediction process as a cooperative game where features act as players contributing to the final prediction. The Shapley value of a feature represents its average marginal contribution across all possible subsets of features. This approach accounts for feature interactions, providing a more nuanced understanding of how each feature affects the model's output.

2.8.3 Mathematical Framework

Consider a set of $N = \{1, ..., n\}$ players forming coalitions (following [42] and [87]). The game's value function $v(S)$ assigns a numerical value to each subset S of players, representing the worth of that coalition. The Shapley value $\varphi_i(v)$ for a player i is defined

as their average marginal contribution across all possible subsets of players:

$$\begin{aligned}\varphi_i(v) &= \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!\,(n-|S|-1)!}{n!}(v(S \cup \{i\}) - v(S)) \\ &= \frac{1}{n} \sum_{S \subseteq N \setminus \{i\}} \frac{1}{\binom{n-1}{|S|}}(v(S \cup \{i\}) - v(S))\end{aligned} \tag{11}$$

where:

- n is the total number of players.
- S is any subset of players from $N \setminus \{i\}$, i.e., any subset that does not include player i.
- $\varphi_i(v)$ represents the value of the coalition S.
- $v(S \cup \{i\})$ represents the value when player i joins S.

Properties of Shapley Values Shapley values satisfy several desirable properties that make them a particularly suitable measure for feature attribution [88]:

- **Efficiency:** The total of individual contributions is equal to the team's realized value (grand coalition).
$$\sum_{i \in N} \varphi_i(v) = v(N) \tag{12}$$
- **Symmetry:** If two players i and j are interchangeable in the sense that
$$v(S \cup \{i\}) = v(S \cup \{j\}) \tag{13}$$
for every subset S of N that does not contain i or j, then they must receive the same Shapley value $\varphi_i(v) = \varphi_j(v)$.
- **Linearity:** If two coalition games described by gain functions v and w are combined, then the distributed gains should correspond to the gains derived from v and the gains derived from w:
$$\varphi_i(v + w) = \varphi_i(v) + \varphi_i(w) \tag{14}$$
for every i in N. Also, for any real number a,
$$\varphi_i(av) = a\varphi_i(v) \tag{15}$$
for every i in N.
- **Null player:** The Shapley value $\varphi_i(v)$ of a null player i in a game v is zero. A player i is *null* if
$$v(S \cup \{i\}) = v(S) \tag{16}$$
for all coalitions S that do not contain i.

By treating features as players in a cooperative game, the Shapley value method provides a robust and theoretically grounded approach for interpreting model predictions. Here, the players correspond to input features, and the game represents the prediction process. The value function $v(S)$ corresponds to the model's prediction when only the features in S are considered. The Shapley value then quantifies how much each feature contributes to the final prediction, averaged across all possible feature subsets. This makes it a valuable tool in explainable AI and feature importance analysis.

2.8.4 Practical Computation of Shapley Values

The exact calculation of Shapley values, as shown in Equation (11), requires evaluating the model on all possible subsets of features, leading to exponential complexity. This makes exact computation infeasible for high-dimensional datasets. To mitigate the computational burden, SHAP (Shapley Additive Explanations) [86] introduces approximation methods, including KernelSHAP:

KernelSHAP : A Model-Agnostic Approximation Kernel SHAP is a model-agnostic method that estimates Shapley values using a weighted linear regression approach [86]. This method allows for the interpretation of any machine learning model without requiring modifications to its structure. It approximates the explanation as a linear function:

$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z'_i, \tag{17}$$

where:

- $g(z')$ is the approximated model output,
- ϕ_0 is the expected model output when no feature is included (*base value*),
- ϕ_i are the Shapley values to be estimated,
- $z'_i \in \{0, 1\}$ indicates whether feature i is present in a given subset.

To ensure a fair approximation, Kernel SHAP assigns a weight to each subset S based on the SHAP kernel:

$$\pi(S) = \frac{(N-1)}{\binom{N}{|S|}|S|(N-|S|)}, \tag{18}$$

where:

- N is the total number of features
- $|S|$ is the number of features included in the subset S
- $N - |S|$ is the number of features excluded from $|S|$

This weighting scheme ensures that subsets of intermediate size contribute more to the estimation process, aligning with the theoretical foundation of cooperative game theory. Kernel SHAP estimates Shapley values by solving a weighted least squares regression problem:

$$\hat{\phi} = \arg\min_{\phi} \sum_{S} \pi(S) \left(f(S) - g(z')\right)^2 . \tag{19}$$

This optimization ensures that $g(z')$ closely approximates the original model f, enabling the extraction of meaningful Shapley values. While KernelSHAP significantly reduces the computational cost compared to exact methods, it remains expensive for models with a large number of features, as it relies on Monte Carlo sampling.

Generating Shapley Values Using Kernel SHAP The SHAP library implements Kernel SHAP as a model-agnostic method for estimating Shapley values. To compute these values, an explainer object must first be created. This explainer corresponds to $g(z')$ from Equation (17) and serves as an approximation of the model's behavior. It estimates the contribution of each feature to the predictions by leveraging a set of reference samples. The process involves the following steps:

1. **Selecting a background dataset** : a subset of instances from the training set is chosen as a reference dataset to define a baseline for feature contributions.

2. **Defining a custom prediction function** : this function represents the model to be explained, which could be a classical neural network (NN) or a quantum neural network (QNN).

3. **Initializing the explainer** : the KernelSHAP explainer is created using the custom prediction function and background samples. This step provides the base value ϕ_0 (the expected model output for the reference dataset) and constructs the approximation function $g(z')$.

4. **Computing Shapley values** : the explainer samples different feature subsets, evaluates their impact on the model's predictions, and assigns Shapley values to each feature.

5. **Generating a SHAP Explanation object** : the final output includes the Shapley values, the expected model output, the test data, and feature names.

2.8.5 Interpretation of Shapley Value Results

Once Shapley values have been computed, their interpretation is crucial for extracting meaningful insights from machine learning models. Several visualization techniques are commonly used to analyze the impact of features on model predictions.
Various plots aid in understanding the distribution and effect of feature contributions, among them the plots that are going to be used in this study will be:

- **Beeswarm Plots:** Beeswarm plots provide an overview of how different features influence model predictions across all samples [86]. Each point represents a Shapley

value for a given instance and feature, with color encoding the feature value. The spread of points along the x-axis indicates the variability of feature importance.

- **Summary Plots (Bar Plots):** A summary plot aggregates Shapley values across the dataset to show the average absolute contribution of each feature to model predictions. Features are ranked by importance, helping to identify key drivers of the model's decisions [42].
- **Dependence Plots:** Dependence plots illustrate the relationship between a feature's value and its corresponding Shapley value. They provide insight into how a feature's magnitude influences predictions and can reveal potential interactions with other features.

3 Methods and Data

In this section, we provide a detailed description of the neural networks used in this study, including their architectures and training procedures. We then introduce the dataset employed for training and evaluation. Furthermore, we outline the evaluation metrics used to assess model performance. Finally, we explain how Shapley values are computed to interpret the models' predictions.

3.1 Classical Neural Network

The classical model used in this study is a feedforward neural network designed for regression tasks. It is designed to learn the relationship between input variables and cloud cover. This model has been developed by Lorenzo Pastori in [35].
The network follows a standard multi-layer architecture, consisting of (Fig. 8):

- An input layer with $N = 6$ neurons, corresponding to the selected atmospheric features.
- Multiple hidden layers, each containing a varying number of neurons activated by a non-linear function (in this case $tanh$) to capture the complexity of the problem.
- An output layer with a single neuron, activated by a linear function to predict cloud cover.

Table 1 summarizes the key characteristics of the classical neural network.

NN	Hidden layers	D	Input features
NN^6	$8 \to 3 \to 7$	119	$\{hus, clw, cli, ta, pa, hwind\}$

Table 1: Specifications of the classical neural network. The notation NN^6 denotes a network with an input layer of 6 neurons, followed by three hidden layers containing 8, 3, and 7 neurons, respectively. D represents the total number of trainable parameters in the network. The input features consist of six atmospheric variables: specific humidity (hus), cloud liquid water content (clw), cloud ice water content (cli), air temperature (ta), pressure (pa), and horizontal wind speed (hwind).

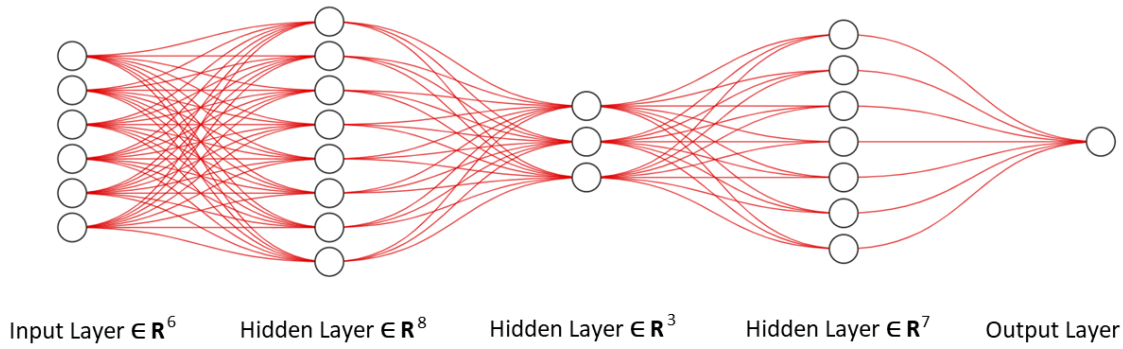

Figure 8: Schematic of NN architecture. The input layer has $N = 6$ neurons corresponding to the 6 atmospheric input features. After the hidden layers there is the output layer with a single node representing the cloud cover prediction.

The network is trained using the mean squared error (MSE) loss function.
For parameter optimization, the Adam optimizer is used. The learning rate is set to 0.001, and the model is trained for 100 epochs with a batch size of 100. Finally, all computations and model implementations are performed using TensorFlow [89].

3.2 Quantum Neural Network

The Quantum Neural Network (QNN) used in this study, also taken from [35], is based on a parameterized quantum circuit (PQC) and designed to predict cloud cover based on six atmospheric features.
The QNN can be broken down into the following stages:

- **Data Encoding** : The QNN uses a qubit register initialized in the $|0\rangle$ state. The input features are encoded using the data re-uploading technique [90], where each input feature is encoded multiple times ($n_{enc} = 4$) using single-qubit rotation gates. This approach increases the number of Fourier frequencies the model can capture, enhancing its ability to represent the input data. The number of qubits used in this stage $N = 6$ corresponds to the number of input features. The encoding layer is defined as:

$$\hat{S}(x) = \prod_{n=1}^{N} e^{-i\frac{x_n}{2}\hat{\sigma}_{\alpha n}}, \tag{20}$$

  where x_n is the n-th component of the vector of input features x, and $\alpha = x, y, z$ denotes the rotation axis, depending on the chosen ansatz.

The encoding process is interleaved with variational blocks $\hat{V}(\vartheta^{(k)})$ $(k = 1, \ldots, n_{\text{enc}})$, which depend on trainable parameters $\vartheta^{(k)}$ and contain entangling operations. The specific form of these blocks is detailed below.

- **Variational Quantum circuit (PQC)** : After the data encoding steps, the model applies $n_{\text{var}} = 2$ variational layers $\hat{W}(\varphi^{(\ell)})$ $(\ell = 1, \ldots, n_{\text{var}})$. These blocks increase the number of trainable parameters and include entangling operations.
  The full parameterized quantum circuit (PQC) is described by a unitary operator that is a product of two components: the variational blocks and the encoding layers. Each component depends on its respective set of trainable parameters that are optimized during training:

$$\hat{U}_{\vartheta,\varphi}(x) = \prod_{\ell=1}^{n_{\text{var}}} \hat{W}(\varphi^{(\ell)}) \prod_{k=1}^{n_{\text{enc}}} \big(\hat{V}(\vartheta^{(k)})\hat{S}(x)\big). \tag{21}$$

- **Measurement and Output Layer** After the PQC computation, the expectation values of the Pauli-Z operators are measured on all qubits. The final output of the QNN is obtained as a weighted sum of these expectation values, with trainable weights and a bias term :

$$f_\theta(x) = b + \sum_{n=1}^{N} w_n \langle \hat{\sigma}_z^n \rangle_{\vartheta,\varphi}(x) \tag{22}$$

  where ϑ represents the set of trainable parameters in the encoding layers, φ represents the set of trainable parameters in the variational blocks, w_n are the weights applied to the measured expectation values, and b is a bias term. The parameter set θ encompasses all these trainable components (ϑ, φ, w, and b).
  After optimization, the final output approximates the classical target function, which corresponds to the cloud cover.

The key characteristics of the QNN are summarized in Table 2:

QNN	N	n_{enc}	n_{var}	D	Input features
QNN^6	6	4	2	109	$\{hus, clw, cli, ta, pa, hwind\}$

Table 2: Specifications of the quantum neural network. The notation QNN^6 denotes a QNN with $N = 6$ qubits, each corresponding to an input atmospheric feature. n_{enc} represents the number of encoding blocks, while n_{var} indicates the number of variational blocks. D denotes the total number of trainable parameters. The input features include six atmospheric variables: specific humidity (hus), cloud liquid water content (clw), cloud ice water content (cli), air temperature (ta), pressure (pa), and horizontal wind speed (hwind).

In order to make a fair comparison, the number of parameters of the two networks is kept comparable.

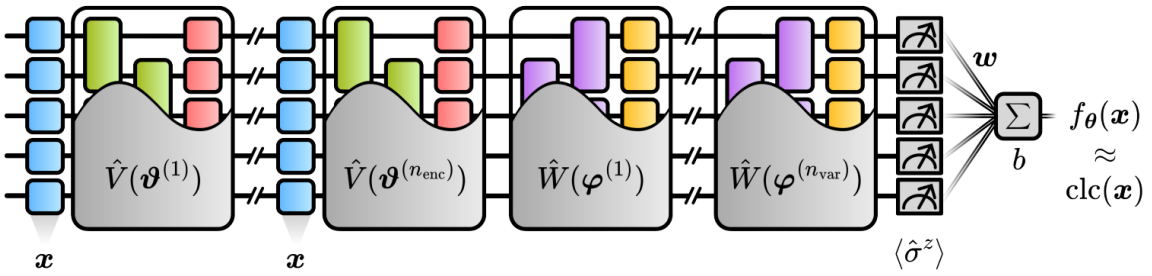

Figure 9: Schematics of QNN architecture. The data $\mathbf{x}$ is uploaded n_{enc} times as angles of single-qubit rotations (blue boxes). In our implementation, each input feature is uploaded to the same qubit each time. These re-uploading gates are interleaved with variational blocks $\hat{V}(\vartheta^{(k)})$ containing entangling gates and trainable parameters $\vartheta^{(k)}(k = 1, ..., n_{enc})$. Afterwards, a sequence of n_{var} variational blocks $\hat{W}(\varphi^{(l)})(l = 1, ..., n_{var})$ are applied. In the end, the expectation values of $\hat{\sigma^z}$ on all qubits are measured, and a weighted average of those is performed, with trainable weights $\mathbf{w}$ and a bias term b. The result $f_\theta(\mathbf{x})$ should approximate the cloud cover $clc(\mathbf{x})$ after training the parameters $\theta = \{\{\vartheta^{(k)}\}_k, \{\varphi^{(l)}\}_l, \mathbf{w}, b\}$. Taken from: [35].

XYZ ansatz For the XYZ circuit ansatz used in this thesis, the encoding blocks take the following form:

$$\hat{V}_{XYZ}(\vartheta) = \hat{R}_{yy}(\vartheta_{(2N-1)\to(3N-3)})\hat{R}_{xx}(\vartheta_{N\to(2N-2)})\hat{R}_{zz}(\vartheta_{1\to(N-1)}), \tag{23}$$

where

$$\hat{R}_{\alpha\alpha}(\vartheta) = \prod_{n=1}^{N-1} e^{-i\frac{\vartheta_n}{2}\hat{\sigma}_{\alpha_n}\hat{\sigma}_{\alpha_{n+1}}}, \quad \text{with } \alpha = x, y, z, \tag{24}$$

and $\vartheta_{i\to j}$ denotes the slice of ϑ from the i-th to j-th component.
The variational blocks for the XYZ ansatz read as:

$$\hat{W}_{XYZ}(\varphi) = \hat{R}_x(\varphi_{(3N-2)\to(4N-3)})\hat{R}_{yy}(\varphi_{(2N-1)\to(3N-3)})\hat{R}_{xx}(\varphi_{N\to(2N-2)})\hat{R}_{zz}(\varphi_{1\to(N-1)}). \tag{25}$$

Optimization and Training : The training of the parameters θ is done via a quantum-classical feedback loop. In each iteration, the QNN is run on the quantum device with the current parameters, and the cost function is computed. The value of this function is then used to propose new parameters that will be used in the next iteration. In this thesis, the computations of the QNN, which would typically take place on a quantum device, are simulated numerically using Pennylane library [91].
For training, the cost function that is minimized is the mean squared error (MSE) calculated over the training dataset.
The parameters θ are updated using gradient descent methods, specifically the Adam optimizer.

3.3 Training Data

Despite their structural differences, both networks have been trained using the same approach and the same data.
The training data used in this work is obtained from global storm-resolving ICON simulations performed as part of the DYnamics of the Atmospheric general circulation Modeled On Non-hydrostatic Domains (DYAMOND) project [92]. These simulations offer an improved representation of clouds and convection compared to simulations at coarser resolutions. The project's first phase ("DYAMOND Summer") included a simulation starting from 1 August 2016 [92], while the second phase ("DYAMOND Winter") was initialized on 20 January 2020 [93]. In both phases, the ICON model simulated 40 days, providing three-hourly output on a grid with a horizontal resolution of 2.47 km. In both cases, the first 10 days have been discarded as spin-up time of the simulation, to have training and testing datasets more closely representing physically realistic conditions. Following [29] we define a high-resolution grid cell to be cloudy (cloud cover = 1) whenever a meaningful cloud condensate (cloud water or cloud ice) amount is detected (i.e., when specific cloud condensate content exceeds 10^{-6} kg/kg) and to otherwise be cloud-free (cloud cover = 0). Such a binary setting of cloud cover is much more sensible at the high horizontal and vertical resolution of the storm-resolving model simulations than at coarser resolutions. Following the methodology of [30], DYAMOND data have been coarse-grained to an ICON grid with a typical climate model horizontal grid resolution of $\approx$ 80 km (corresponding to an R2B5 ICON grid typically used in climate projections). Vertically, data have been coarse-grained from 58 to 27 layers below an altitude of 21 km, which is the maximum altitude with clouds in the data set. After coarse-graining, cloud cover in a given cell can take any value between 0 and 1, representing the fraction of the cell that is occupied by clouds. Given that cloud cover cannot exist in the absence of cloud condensate, all the cells where the total amount of cloud condensate is zero are removed from the dataset. This results in a dataset which is more balanced, i.e., where the cloud-free samples are less over-represented. We then split the data into a training and a validation set.
To ensure a diverse representation of atmospheric conditions, the training and test sets are constructed by randomly sampling data points. The training set consists of 100,000 samples, and model evaluation is conducted on an independent test set of equal size, ensuring no data leakage between the two subsets.

3.3.1 Input Features

Both classical and quantum models receive as input six key atmospheric variables for each data sample, selected for their relevance in cloud cover formation [35].
These variables are:

- **Specific humidity** (*hus*) [kg/kg]
- **Cloud liquid water content** (*clw*) [kg/kg]
- **Cloud ice conten**t (*cli*) [kg/kg]
- **Air temperature** (*ta*) [K]

- **Pressure** (pa) [Pa]
- **Horizontal wind component** ($hwind$) $= \sqrt{u^2 + v^2}$ [m/s] : magnitude of horizontal wind component (with u and v being the zonal and meridional components, respectively).
- **Altitude** (z_g) [m]
- **Coriolis Force** [N]

3.3.2 Preprocessing

Due to the varying magnitudes and distributions of these features, an appropriate transformation and rescaling are required before feeding them into the models. In the case of the quantum neural network, the input data are encoded as angles, making it convenient to transform the features into the $[0, \pi]$ range. For temperature (ta) and pressure (pa), a simple min-max scaling to the interval $[0, \pi]$ is applied. However, specific humidity, cloud liquid water content, cloud ice content, and horizontal wind exhibit highly skewed distributions, with values concentrated near zero and long decaying tails. To address this, a non-linear logarithmic-like transformation that spreads the values more uniformly while preserving the behavior of the tails is applied.
To enable a direct comparison between quantum and classical neural networks, the same preprocessing steps are applied to both models. Additionally, the cloud cover output undergoes a transformation via a monotonic function g, such that the training targets are given by $y_i = g(clc(x_i))$. This transformation ensures that the output values are approximately uniformly distributed in the interval $[0, 1]$, improving model training stability. All the transformations are explained in Appendix (A).

3.4 Evaluation metrics

To evaluate the performance of the models, the mean squared error (MSE) and the coefficient of determination (R^2) are used. These metrics are computed using the `mean_squared_error` and `r2_score` functions from the `sklearn.metrics` module. The MSE measures the average squared difference between the true values y_i and the predicted values $\hat{y}_i$, and is given by:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{26}$$

where n is the number of samples. Lower MSE values indicate better predictive accuracy. The R^2 score, also known as the coefficient of determination, assesses how well the predictions approximate the true values. It is defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2} \tag{27}$$

where $\bar{y}$ is the mean of the true values. An R^2 score close to 1 indicates that the model explains most of the variance in the data, while a value close to 0 or negative suggests poor predictive performance.

3.5 Shapley Values

To analyze feature importance, Shapley values are computed using the SHAP library [86]. Specifically KernelSHAP has been employed, a model-agnostic method that approximates Shapley values by treating the model as a black box and estimating contributions through perturbations of the input features. This method was aplicable to both NN and QNN.
Since KernelSHAP requires a background dataset to approximate feature attributions, 100 representative samples have been selected from the training dataset using `shap.sample()`, which performs a random stratified sampling. The background dataset serves as a reference distribution for estimating feature contributions.
Once the explainer was initialized, Shapley values have been computed for the entire test dataset (100,000 samples). The Shapley values were obtained using a custom prediction function, which feeds input samples to the trained classical or quantum neural network and returns the predicted outputs.
The computation resulted in a set of Shapley values representing the contribution of each feature to the model's predictions.
For clarity, pseudocode outlining the exact implementation is provided in the appendix.

4 Results

This section presents the results of the study, focusing on the comparison between classical and quantum models for cloud cover parameterization. As discussed in previous sections, the use of machine learning for parameterizations represents a promising alternative to improve climate models. Neural networks are employed to directly predict cloud cover, fully replacing traditional parameterization schemes. The analysis aims to interpret the decision-making process of the networks from a physical perspective. The results of the study will be presented as follows:

- **Models Performance**: The predictive capabilities of neural networks are assessed using standard evaluation metrics, such as the mean squared error (MSE) and the coefficient of determination (R^2).
- **Feature importance Analysis**: Model interpretability will be analyzed through Shapley values, focusing on feature importance rankings and examining whether QML models capture physical relationships similar to those learned by classical counterparts. Additionally, the relationship between Shapley values and corresponding feature values will be explored to gain deeper insights into models behavior.
- **Comparison with Empirical Parametrization:** The feature importance will be compared with the parameterization approach of Xu & Randall [55], which will serve as a benchmark for cloud cover parameterization.
- **Cloud Regimes Analysis**: The generalization capabilities of the models will be investigated by evaluating their performance and feature importance across different atmospheric regimes, assessing their consistency under varying physical conditions.
- **Stability Analysis:** The robustness of classical and quantum models will be examined by analyzing the variability of learned relationships across multiple training runs.
- **Eight Features Networks:** The impact of increasing the number of input features to eight is explored, assessing potential improvements in predictive performance and interpretability.

4.1 Models Performance

As first step of the analysis, the performance of the two networks has been evaluated on a test dataset. Model predictions are compared against ground truth values using standard regression metrics. The histograms in Figure 10 illustrate the distribution of predicted cloud cover values compared to the ground truth:

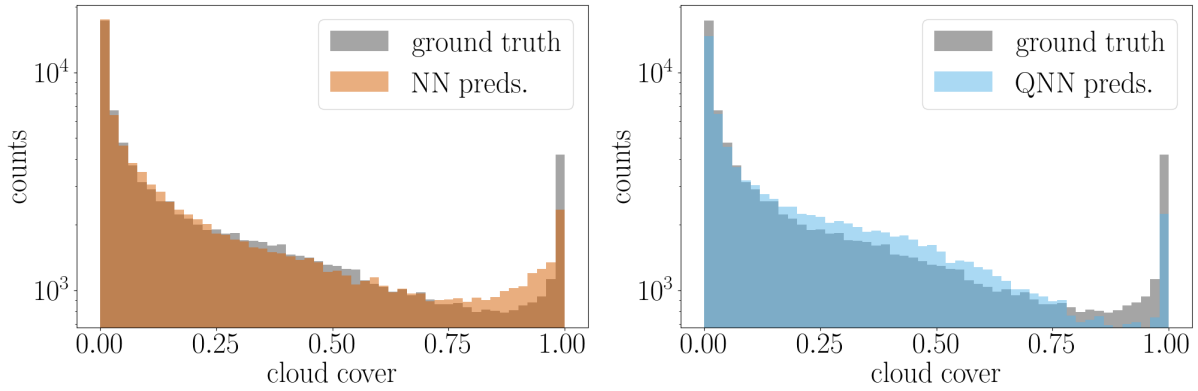

Figure 10: Comparison of histograms: Classical NN (left) and Quantum NN (right). The x-axis represents cloud cover (ranging from 0 to 1), divided into 50 bins, while the y-axis shows frequency counts for each bin, based on a sample of 100,000 test points.

Table 3 summarizes the MSE and R^2 values for both models:

Model	**MSE**	$\mathbf{R^2}$
Classical	0.006	0.94
Quantum	0.011	0.89

Table 3: Performance metrics for Classical and Quantum Neural Networks.

Both models perform well, with the classical neural network achieving slightly better accuracy, as indicated by lower MSE and higher R^2. Despite this, both networks successfully predict cloud cover, although each exhibits some biases:

- The classical model slightly overestimates cloud cover in the 0.8 – 0.9 range.
- The quantum model, on the other hand, tends to underestimate cloud cover in the same range but exhibits slight overestimation in the 0.2 – 0.6 interval.
- Both models underestimate the occurrence of fully cloud-covered cells (1.0).

These differences likely stem from the distinct architectures of the two models and their respective mechanisms for processing information. To gain deeper insights into these variations, we next analyze how each model prioritizes and utilizes input features through an examination of Shapley values.

4.2 Feature importance analysis

Beyond evaluating the overall predictive performance of the models, it is important to assess whether the networks are effectively learning the underlying physical processes governing cloud cover formation. Neural networks, with their capacity to model highly non-linear relationships, are well-suited for tasks such as cloud cover parametrization. However, their black-box nature presents a significant challenge when attempting to interpret their decision-making process, particularly in scientific applications where physical consistency is crucial.
To address this issue, we employ methods from Explainable AI (XAI), which provide tools for interpreting the internal workings of machine learning models. Among these, Shapley values offer a principled approach to quantifying feature importance by attributing a contribution to each input variable based on its marginal impact on the model's predictions. In the context of cloud cover prediction, Shapley values allow us to examine which features are most influential in determining the cloud cover. This allows us to examine whether the features identified as most influential align with established physical mechanisms of cloud formation and to see how the value of the feature impact the prediction. By analyzing feature importance, we aim to gain a clearer understanding of how the models reflect known atmospheric processes and whether their learned patterns align with physical intuition.
To visualize and analyze these feature contributions, we use at first a beeswarm plot, which provides a clear graphical representation of Shapley values. Examining these plots for both the classical and quantum neural networks allows us to compare their feature importance attribution.

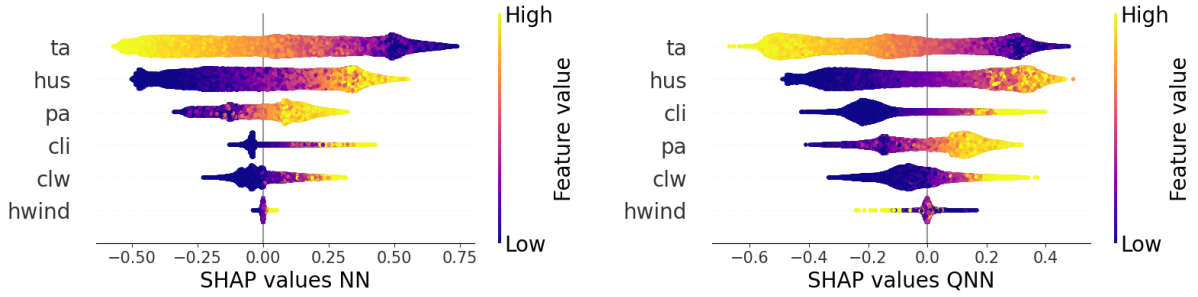

Figure 11: Beeswarm plots of SHAP values for the classical neural network (left) and the quantum neural network (right). Each plot is generated using 100,000 data points. The features are ranked by their average absolute Shapley values, with the most influential features appearing at the top. Each point represents a single data instance, with thicker clusters indicating a higher density of points. The color represents the feature value, ranging from low (purple) to high (yellow).

Figure 11 presents the beeswarm plots for the classical and quantum neural networks, respectively. In both plots, features are ordered according to their importance, as determined by SHAP values. The ranking is calculated by averaging the absolute SHAP values for each feature across all instances in the dataset, reflecting the overall contribution of each

feature to the model's predictions. Features with higher average absolute SHAP values are positioned at the top, indicating their greater influence on the model. Upon analyzing the plots, we observe that the most important feature for both the classical and quantum models is temperature, followed by specific humidity. However, some differences emerge in the ranking of secondary features. A detailed analysis of each feature is presented below:

- **Temperature:** The Shapley values indicate a clear trend: lower temperatures correspond to positive values, suggesting increased cloud cover, while higher temperatures result in negative values, implying reduced cloud formation. This aligns with the physical understanding that higher temperatures enhance moisture-holding capacity, inhibiting condensation.

- **Specific Humidity:** As discussed in Section 2, cloud cover strongly depends on humidity. The Shapley values confirm this: high specific humidity leads to positive values, indicating increased cloud cover, while drier conditions correspond to negative values, reflecting reduced cloud formation.

- **Pressure:** Lower pressure, typically found at higher altitudes, is associated with a decreased probability of cloud formation, as indicated by the negative Shapley values. This aligns with the fact that lower air density at high altitudes makes condensation less favorable.

- **Cloud Ice:** The quantum model assigns greater importance to cloud ice, ranking it as the third most influential feature. The Shapley values highlight its role in cloud formation: higher cloud ice levels contribute positively, while lower levels or absence of condensate lead to negative contributions.

- **Cloud water:** Like cloud ice, cloud water positively influences cloud cover predictions, as reflected in the Shapley values. However, in both models, it is less influential than cloud ice.

- **Horizontal wind:** The distribution of Shapley values around zero confirms that horizontal wind has minimal impact on cloud cover predictions.

At a first impact, overall, the feature importance align with our physical expactations. While some differences are observed (such as the higher ranking of cloud ice in the quantum model) these variations may arise from differences in how the two architectures encode and process information.
Now we want to zoom into each feature and see how the Shapley values are distributed with the feature values.

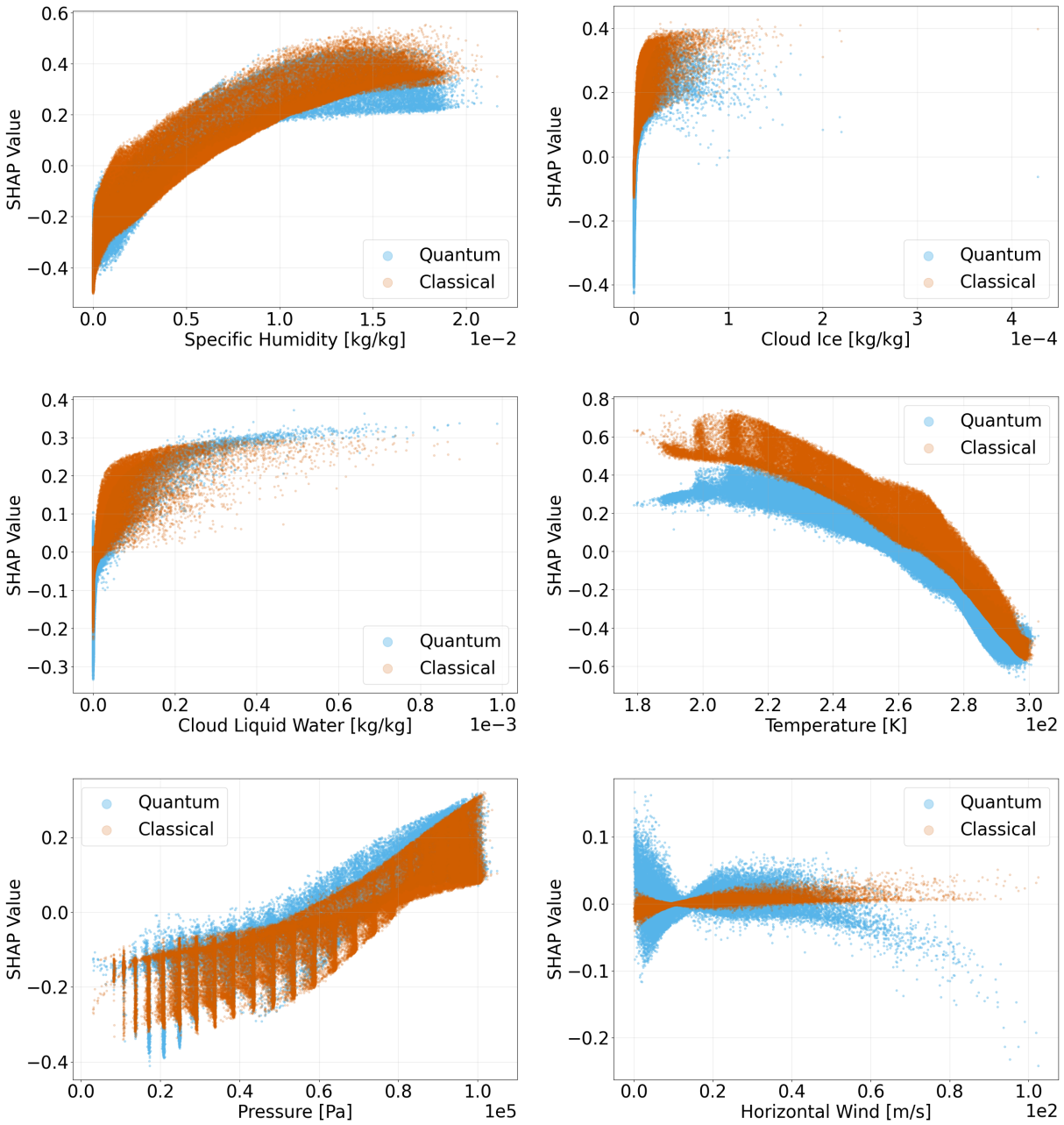

Figure 12: Shapley value distribution for every atmospheric feature. Each plot contains 100,000 data points. The x-axis represents the feature values, while the y-axis shows the associated Shapley values, indicating the contribution of each feature to the model's predictions. We overlap classical (orange) and quantum (blue).

Figure 12 presents the Shapley value distribution for each feature with respect to its corresponding values. The plots for the classical and quantum models are overlapped to facilitate visual comparison and highlight potential differences in how each model interprets the feature importance. Below, we analyze the key trends observed for each feature.

- **Specific Humidity:** Both models exhibit an increasing trend, with higher humidity corresponding to higher Shapley values. This confirms that both networks recognize

the positive correlation between specific humidity and cloud formation, in line with our physical expectation.

- **Cloud Ice and Cloud Water:** The SHAP values for both features exhibit a threshold effect. Below a certain condensate level, the values remain close to or below zero, while higher condensate amounts lead to strong positive contributions to cloud cover predictions. This is expected, as cloud water and ice are fundamental components of cloud formation.
  The differences observed in the beeswarm plot rankings now become clearer: in the quantum model, cloud ice appears more important than in the classical case. This is also reflected in the Shapley value distribution: at low condensate values, the Shapley values for cloud ice are significantly more negative in the quantum model, while in the classical model, they remain closer to zero. This suggests that the quantum network assigns a much lower cloud cover when cloud ice levels are low. A similar but less pronounced pattern is observed for cloud water.

- **Temperature:** As the most influential feature, temperature exhibits a clear decreasing trend in its Shapley value distribution: higher temperatures correspond to increasingly negative Shapley values (indicating reduced cloud cover), while lower temperatures are associated with positive Shapley values (indicating increased cloud cover). This aligns with the physical understanding that lower temperatures promote condensation, facilitating cloud formation.
  The importance of temperature as the dominant feature is further supported by the range of Shpley values along the y-axis, which is broader compared to other features, indicating a stronger influence on model predictions.

- **Pressure:** The Shapley values for pressure confirm its role as a secondary but still relevant factor. The feature distribution exhibits distinct peaks, which correspond to the discrete vertical layers in the atmospheric model. The Shapley values suggest that lower altitudes (higher pressure) are associated with increased cloud cover, while higher altitudes (lower pressure) suppress it. This behavior is consistent with physical expectations, as higher altitudes are characterized by thinner, drier air, which inhibits condensation.

- **Horizontal Wind:** As expected, horizontal wind exerts minimal influence on cloud cover predictions, as indicated by its near-zero Shapley values across most of its range. However, while the classical model assigns consistently low importance to this feature, the quantum model exhibits a more dispersed Shapley value distribution.

A key observation from these plots is that the Shapley value distributions are comparable for both networks across all features. This suggests that, despite their different architectures, both models learn similar patterns in the data and identify the same key drivers of cloud formation.
While minor differences exist (such as the different ranking of cloud ice (cli) in feature importance) the overall behavior of the networks remains qualitatively similar. This is confirmed by the distributions in Figure 12, where the shape of the trends remains unchanged, indicating that cli influences the predictions in a comparable manner despite

its different importance ranking.
Finally, the Shapley values not only reveal a similar behavior between the two networks in terms of distribution shape but also in magnitude, further supporting the idea that both architectures have learned comparable relationships between input features and cloud cover.

4.3 Comparison with empirical parametrization

To better understand the physical consistency of the neural networks learning results, it is useful to compare them with an empirical parameterization that captures physical relationships from data through explicit equations. In this analysis, we will use the Xu-Randall parameterization [55], which has been described in Section 2. This parameterization estimates cloud fraction (CLC_{XR}) as a function of five atmospheric variables: temperature, specific humidity, pressure, cloud ice, and cloud water (which are the same used by the networks excluding horizontal wind). The parameterzation has been implemented as follows:

$$CLC_{XR} = \left(1 - e^{-a(clw + c \times cli)}\right) RH^b \tag{28}$$

where:

- clw is the specific cloud liquid water content,
- cli is the specific cloud ice water content,
- RH is the relative humidity,
- a, b, c are empirical parameters that need to be calibrated.

The parameters a, b, c were optimized by minimizing the Mean Squared Error (MSE) between predicted and observed cloud cover values:

$$MSE = \frac{1}{N} \sum_{i=1}^{N} \left(CLC_{XR}^{(i)} - CLC_{obs}^{(i)}\right)^2 \tag{29}$$

where N is the number of data points (100,000). The optimization was performed using the Nelder-Mead method, with initial values randomly sampled within predefined bounds:

$$1.0 \times 10^4 \leq a \leq 1.0 \times 10^5 \tag{30}$$

$$0.9 \leq b \leq 1.1 \tag{31}$$

$$0.5 \leq c \leq 4.0 \tag{32}$$

Multiple iterations were run with different initial conditions, selecting the parameter set that yielded the lowest MSE:

- $a = 1.014 \times 10^5$
- $b = 6.066$

- $c = 2.608$

Although this parameterization is relatively simple, it has been shown to provide accurate cloud cover predictions, as shown in Table 4.

MSE	$\mathbf{R^2}$
0.011	0.88

Table 4: MSE and R^2 values for Xu-Randall parameterization.

This makes it a valuable benchmark for evaluating the performance of both the classical and quantum neural networks. Moreover, its simplicity makes it easy to interpret, which facilitates a direct comparison with the neural networks, since it uses the same atmospheric variables.
While the neural networks are optimized for minimizing MSE, the Xu-Randall scheme is based on heuristic approximations of atmospheric processes. This fundamental difference in approach raises the question of whether the neural networks have learned a physically meaningful representation of cloud cover or merely an empirical mapping based on the training data.
To investigate this, we analyze the feature importance rankings assigned by the Xu-Randall parameterization. Specifically, we compare the mean Shapley values of each feature across the three models using a bar plot, that shows the mean Shapley value for each feature. This visualization allows us to assess the relative impact of each variable on model predictions.

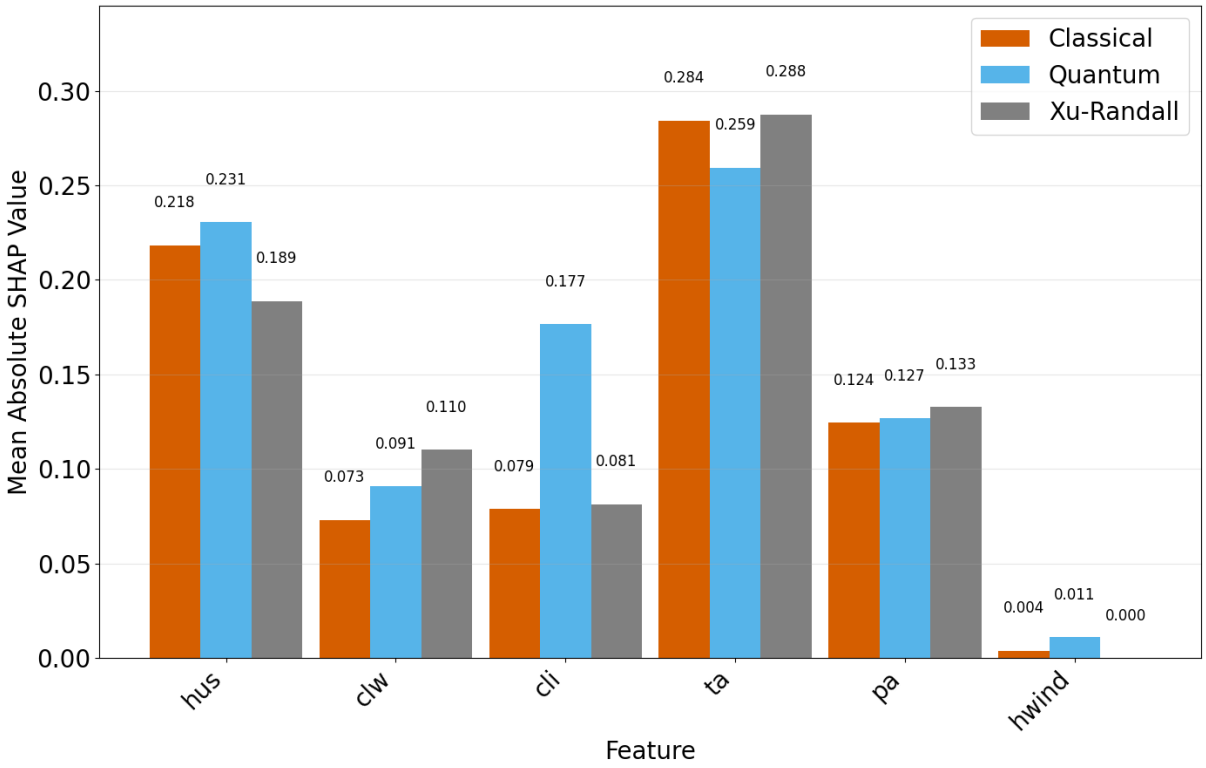

Figure 13: Comparison of mean absolute Shapley Values (classical vs quantum NN vs Xu-Randall scheme). The x-axis represents the input features and the y-axis shows the mean Shapley values. The mean SHAP value has been calculated using 100,000 data points.

From this plot, we can observe an overall agreement between the three models, particularly for the most important variables. This suggests that both the classical and quantum neural networks have successfully identified the key meteorological variables that drive cloud formation, similar to the empirical relationships captured by the Xu-Randall parameterization. The most significant discrepancy is observed with cloud ice, which is given significantly more weight in the quantum model compared to both Xu-Randall and the classical network.
Next, we use a beeswarm plot of Shapley values to visualize the contribution of each input variable to the Xu-Randall predictions.

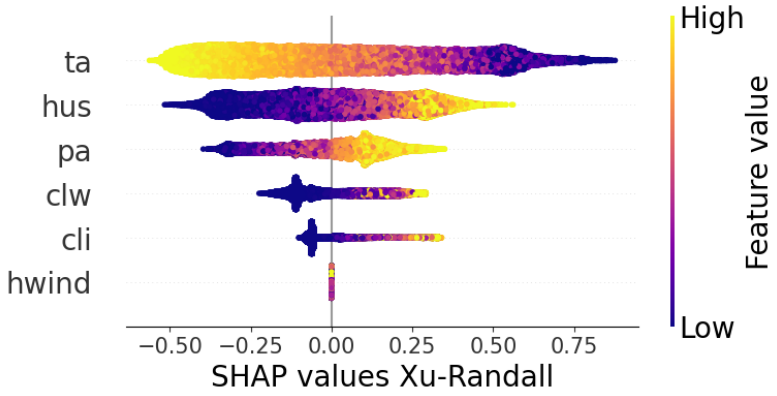

Figure 14: Beeswarm plot for Xu-Randall parametrization. The plot has been generated using 100,000 data points. Each point in the plot represents a data point, and thicker clusters indicate more frequent occurrences. The color gradient represents the value of each feature, ranging from low values (depicted in purple) to high values (depicted in yellow).

The feature ordering in the plot, as well as the relationship between the Shapley values and the corresponding feature values, aligns well with our established physical understanding and is consistent with the behavior observed in the two previously analyzed networks. Specifically, we observe that the Shapley values tend to be higher for lower temperatures, as well as for higher humidity and pressure. Moreover, low values of condensate (both in the form of ice (cli) and liquid water (clw)), correlate with low Shapley values.

4.4 Cloud Regimes

To gain deeper insight into the physical consistency of the neural networks, we analyze their performance across different cloud regimes. Instead of evaluating the models globally, we divide the dataset into distinct atmospheric conditions. This approach allows us to examine how well the models generalize across different meteorological environments and whether they adapt their predictions accordingly. The networks were not retrained for each regime, enabling us to assess their ability to adapt without additional fine-tuning. Clouds can form in a wide range of atmospheric conditions. Here, following [29], we classify the cloud cover into four primary regimes by considering two key physical variables:

- **Air pressure (pa)**
- **Total amount of condensate ($clt = clw + cli$)** : A measure of the total cloud water and cloud ice content

To define meaningful thresholds, we use the median values of air pressure and total condensate in our dataset : $pa = 78,787\ Pa$ and $clt = 1.62 \cdot 10^{-05}$ [kg/kg]. The four resulting regimes are:

1. Low air pressure, little condensate (cirrus-type cloud regime).

2. High air pressure, little condensate (cumulus-type cloud regime).
3. Low air pressure, substantial condensate (deep convective-type cloud regime).
4. High air pressure, substantial condensate (stratus-type cloud regime).

4.4.1 Cirrus-type cloud regime

Cirrus clouds are high-altitude clouds that form at low air pressure and contain little condensate. They are composed mostly of ice crystals and are often thin and wispy.
In the following plots, we compare the performance of the classical and quantum neural networks in this regime.

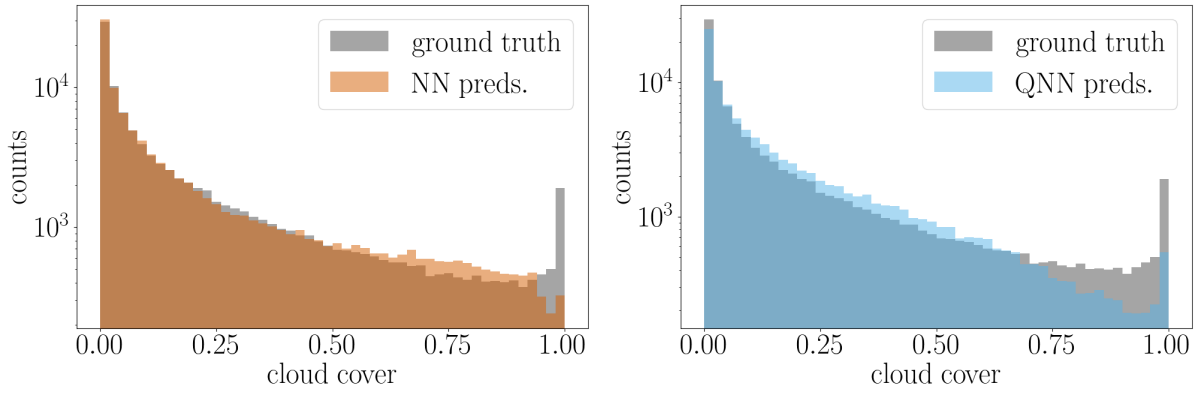

Figure 15: Comparison ground truth vs NN predictions (left) and ground truth vs QNN predictions (right) for the cirrus-type cloud regime. For the evaluation, 100,000 data points have been sampled. The x-axis represents cloud cover (ranging from 0 to 1), divided into 50 bins. The y-axis shows the frequency of occurrences for each bin.

Since we are at high altitude and low condensate levels, the majority of the data points are concentrated around cloud cover values near zero, with the count decreasing as the cloud cover values increase.
The table below provides a quantitative comparison of the models in terms of mean squared error (MSE) and R^2 score.

Model	**MSE**	R^2
Classical NN	0.006	0.92
Quantum NN	0.009	0.86

Table 5: Comparison of MSE and R^2 values for the classical and quantum Neural Networks in cirrus-type regime

Both models achieve good performance, with the classical neural network performing slightly better. However, some notable differences emerge when analyzing their prediction tendencies:

- The classical model tends to overestimate the cloud cover in the 0.7 to 0.9 range.
- The quantum model, on the other hand, tends to underestimate in the same range, but overstimate in the 0.2-0.6 range.

To gain further insight into how the models make predictions, we examine the Shapley values. The following plot compares feature importance for both networks.

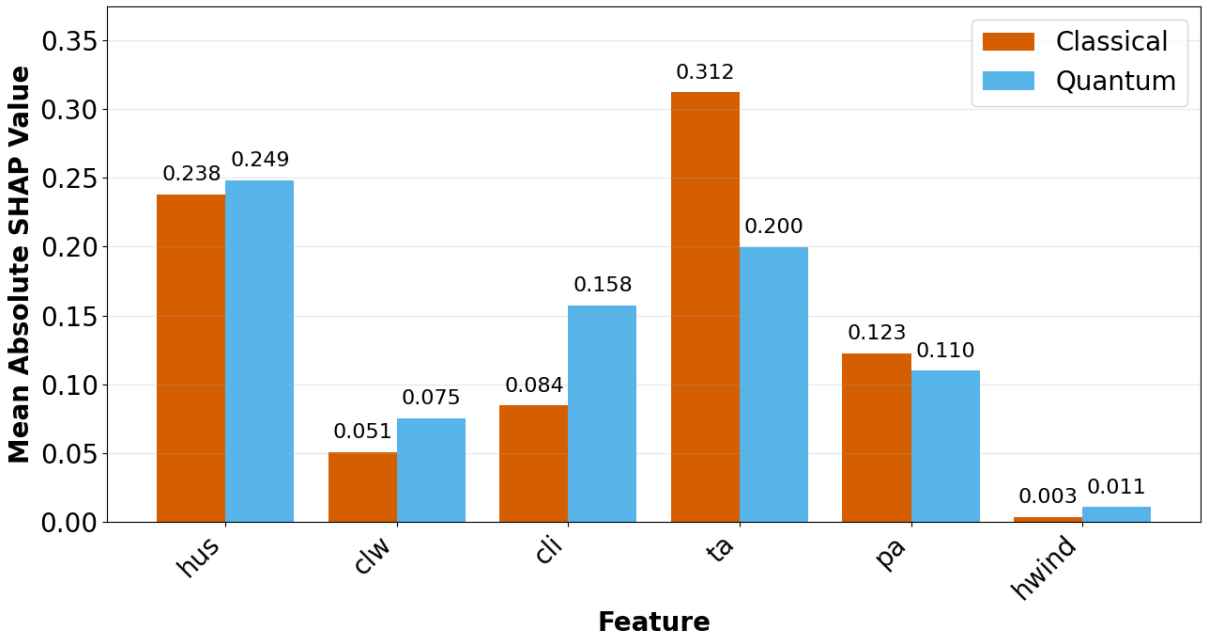

Figure 16: Comparison of feature importance (cirrus-type cloud regime). The x-axis represents the input features, while the y-axis shows the mean Shapley values.

In this regime, the most important features for both networks are temperature and specific humidity, even if the quantum network gives less importance to temperature in comparison to the classical.
Additionally, as already seen in the general case, the quantum network assigns more importance to cloud ice (*cli*) compared to the classical network, which could be worth further investigation.

4.4.2 Cumulus-type cloud regime

Cumulus clouds are typically found at lower altitudes. Below, we compare the performances of the classical and quantum neural networks in this regime.

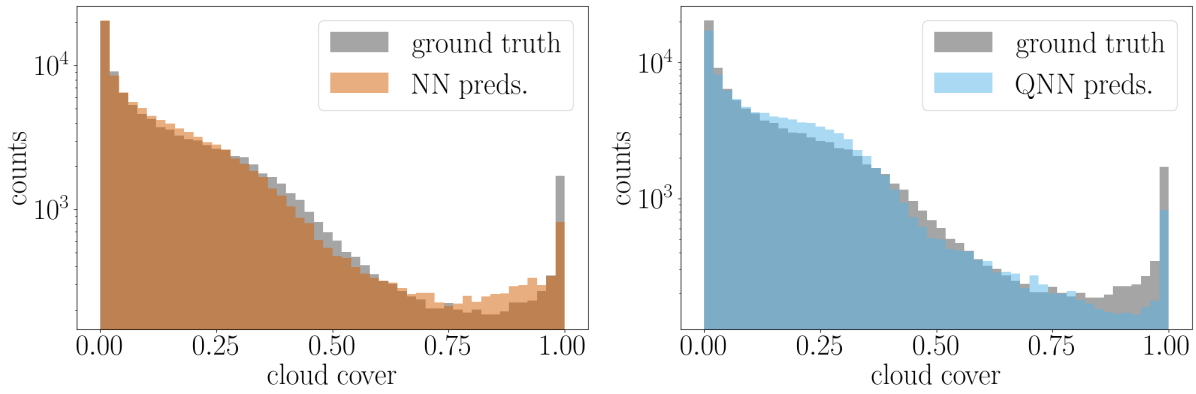

Figure 17: Comparison of ground truth vs NN predictions (left) and ground truth vs QNN predictions (right) for the cumulus-type cloud regime. The evaluation is based on 100,000 data points. The x-axis represents cloud cover (ranging from 0 to 1), divided into 50 bins. The y-axis shows the frequency of occurrences for each bin.

As shown in the histograms, most data points in this regime correspond to low to mid-range cloud cover values (below 0.5), meaning that complete cloud cover is rare also in this regime, aligning with the typical characteristics of cumulus clouds. The following table presents the MSE and R^2 scores for both models:

Model	**MSE**	R^2
Classical NN	0.005	0.91
Quantum NN	0.007	0.86

Table 6: Comparison of MSE and R^2 values for Classical and Quantum Neural Networks in the cumulus-type cloud regime.

Both models exhibit good performance, with the classical neural network yielding slightly better results overall. However, the prediction tendencies differ between the two models:

- The classical network tends to overestimate cloud cover in the 0.8 to 0.9 range.
- The quantum network slightly overestimates in the 0.1 to 0.3 range, while it underestimates for values between 0.8 and 1.0.

Next, we analyze the mean Shapley values in this regime to understand which features the models consider most important.

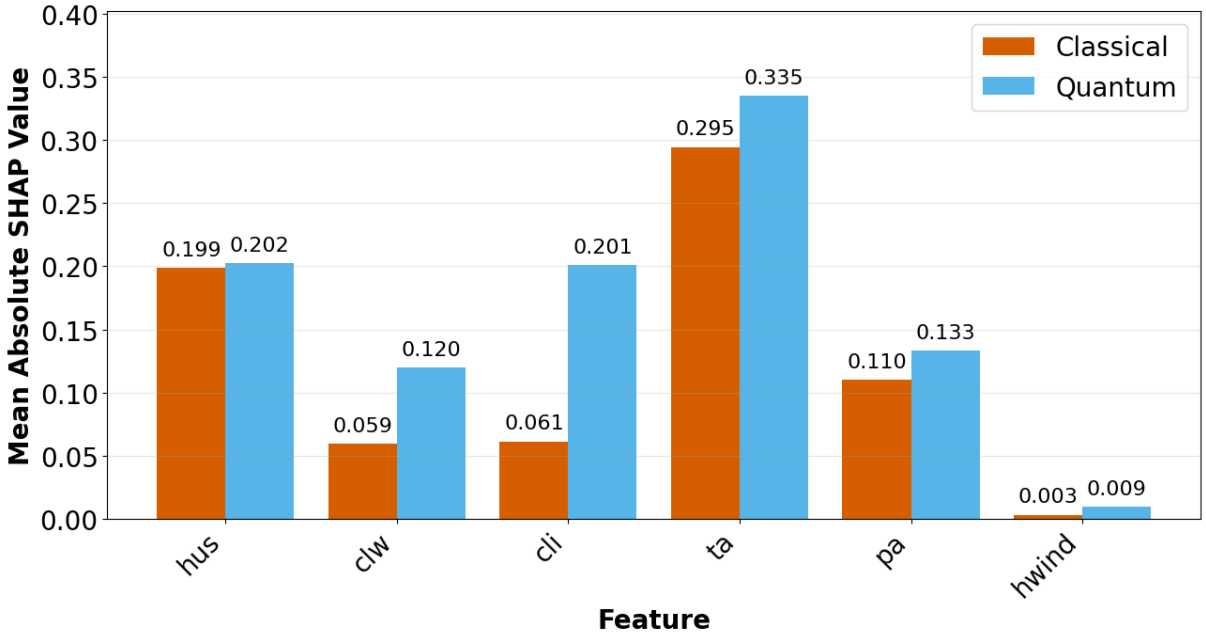

Figure 18: Feature importance comparison for the cumulus-type cloud regime. The x-axis represents the input features, while the y-axis shows the mean Shapley values.

As in the cirrus-type regime, the most important features for both networks are temperature and specific humidity. However, in this case, the quantum network assigns equal importance to cloud ice and specific humidity, suggesting that it relies more heavily on condensate-related features compared to the other regime. This trend is consistent with the broader observation that the quantum model generally attributes more importance to condensate-related variables.

4.4.3 Deep convective-type cloud regime

Deep convective clouds form under conditions of low pressure and substantial condensate. These clouds are typically associated with intense atmospheric instability and result in nearly complete sky coverage [12]. We compare the performance of the classical and quantum neural networks in predicting cloud cover in this regime.

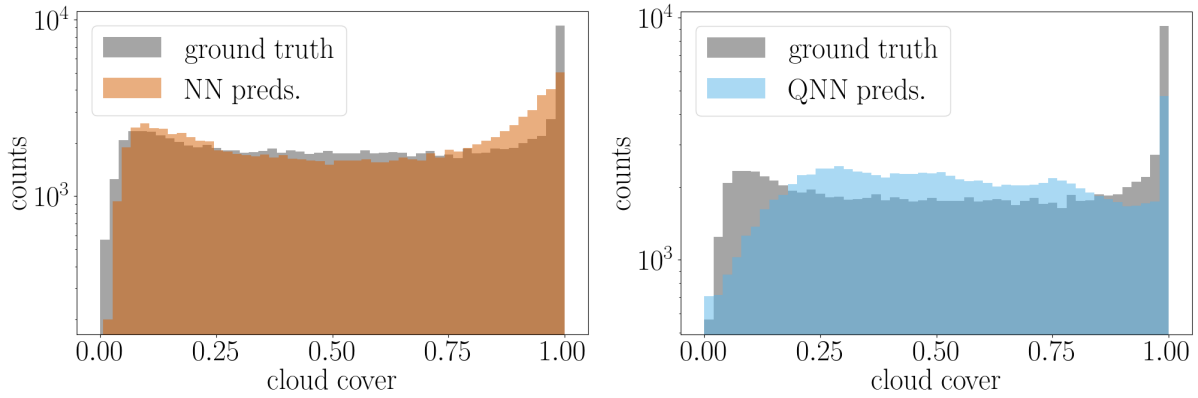

Figure 19: Comparison of ground truth vs NN predictions (left) and ground truth vs QNN predictions (right) for the deep-convective-type cloud regime. The evaluation is based on 100,000 data points. The x-axis represents cloud cover (ranging from 0 to 1), divided into 50 bins. The y-axis shows the frequency of occurrences for each bin.

The distribution of cloud cover values in this regime differs significantly from previous cases:

- There are fewer data points with low cloud cover than in the previously analyzed regimes.
- The distribution is relatively flat in the mid-range values.
- The highest concentration of points is near a cloud cover of 1.0, which aligns with the fact that deep convective clouds frequently lead to nearly complete sky coverage.

The following table presents the MSE and R^2 scores for both the classical and quantum neural networks in this regime:

Model	**MSE**	R^2
Classical NN	0.011	0.89
Quantum NN	0.021	0.79

Table 7: Comparison of MSE and R^2 values for Classical and Quantum Neural Networks in deep convective-type regime.

Both networks exhibit a slight decrease in performance compared to other regimes, with the classical network still outperforming the quantum network. The prediction tendencies show distinct behaviors:

- The classical network alternates between overestimation and underestimation across different cloud cover values.
- The quantum network underestimates cloud cover at the extremes (near 0 and 1) while overestimating in the mid-range values.

We now examine the Shapley values for feature importance.

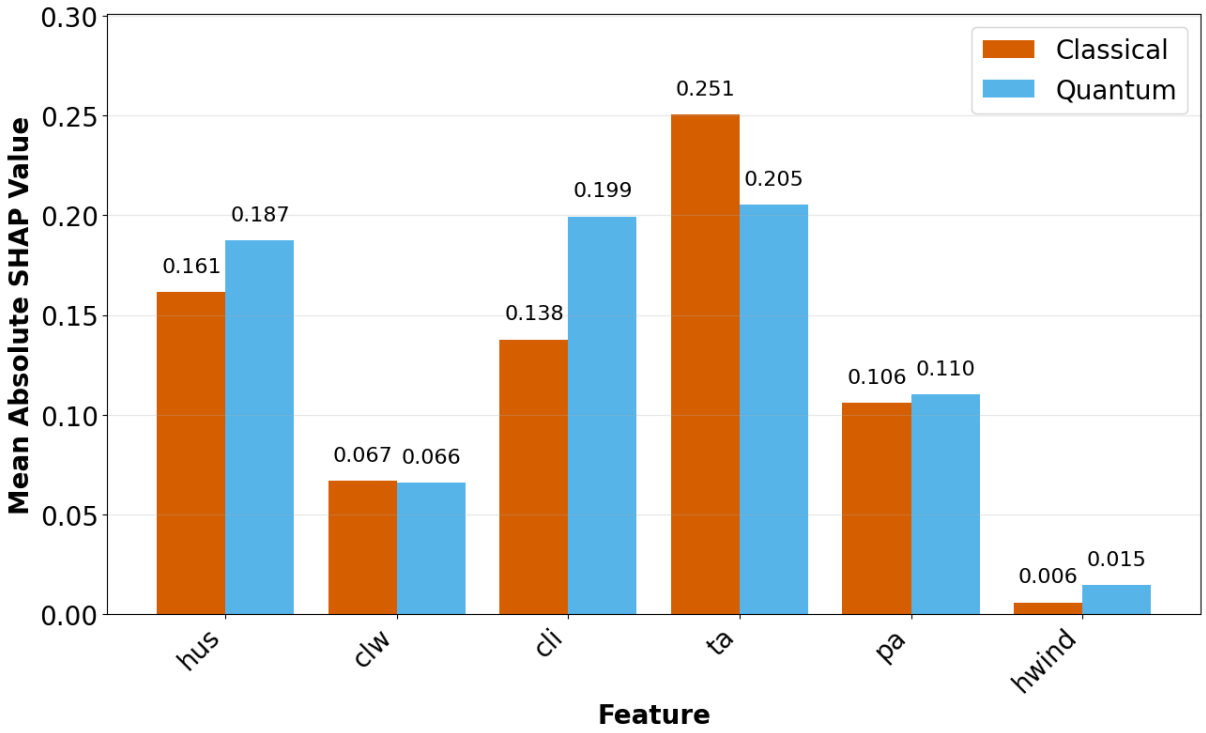

Figure 20: Feature importance comparison for the deep convective-type cloud regime. The x-axis represents the input features, while the y-axis shows the mean Shapley values.

In this regime, the distribution of feature importance differs from previous cases:

- There is less variability in feature importance, meaning that no single feature dominates as strongly as in other regimes.
- Temperature remains the most important feature for both models.
- For the quantum network, cloud ice is more important than specific humidity, suggesting that it relies more on condensate-related information.
- However, both specific humidity and cloud ice play significant roles in both models, with less emphasis on cloud liquid water and air pressure.

4.4.4 Stratus-type cloud regime

Stratus clouds are low-altitude, horizontally extensive formations that develop under high-pressure conditions with substantial condensate, resulting in persistent cloud cover. We now compare the performance of both networks in this regime.

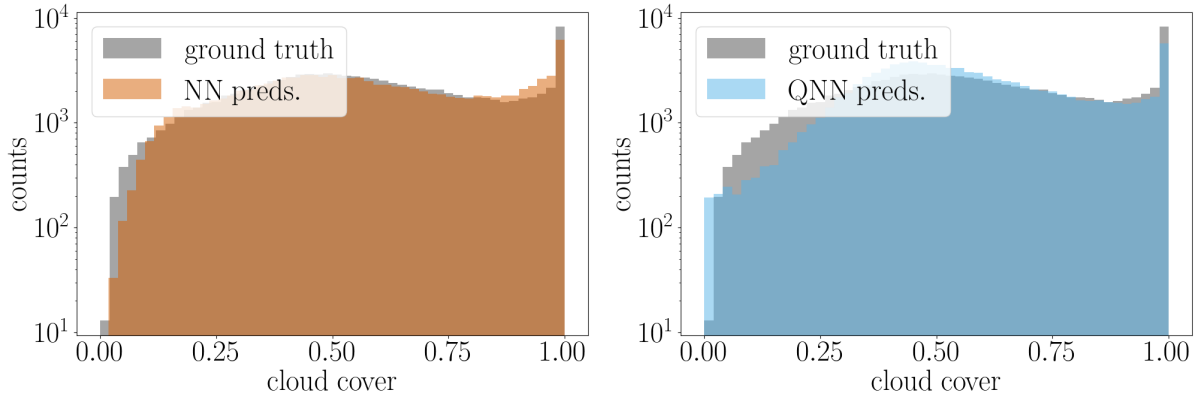

Figure 21: Comparison of histograms: Ground Truth vs NN Predictions (left) and Ground Truth vs QNN Predictions (right) for the stratus-type cloud regime. For evaluation, 100,000 data points were selected. The x-axis represents cloud cover (ranging from 0 to 1), divided into 50 bins. The y-axis shows the frequency of occurrences for each bin.

In this case, cloud-free cells are almost nonexistent, and most data points correspond to high cloud cover values. We summarize the mean squared error and R^2 in Table 8.

Model	**MSE**	R^2
Classical NN	0.006	0.90
Quantum NN	0.013	0.81

Table 8: Comparison of MSE and R^2 values for Classical and Quantum Neural Networks in stratus-type regime

Both models exhibit improved performance compared to the deep-convective regime, with the classical network slightly outperforming the quantum one. The classical network tends to underestimate low cloud cover values and slightly overestimate those near 0.9. The quantum network, on the other hand, alternates between underestimation and overestimation across the range of cloud cover values.
Next, we analyze feature importance.

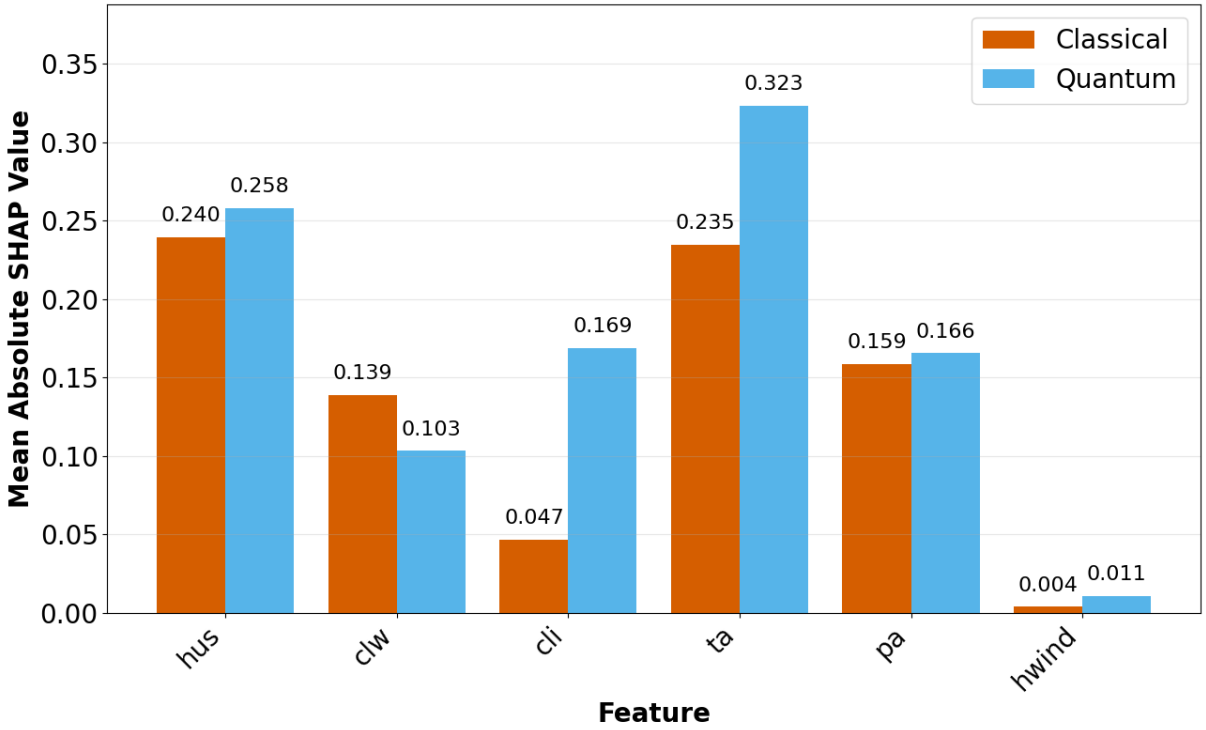

Figure 22: Feature importance comparison for the stratus-type cloud regime. The x-axis represents the input features, while the y-axis shows the mean Shapley values.

In this regime, there is no pronounced difference between the most and least important features, indicating a more homogeneous feature contribution. However, temperature and specific humidity remain the dominant factors for both models. The classical network assigns almost no importance to cloud ice, whereas the quantum network consistently attributes a significant role to it, similar to what was observed in previous regimes.

After analyzing all the different regimes, we can draw the following conclusions:

- The average Shapley value for cloud ice is consistently higher in the quantum network compared to the classical one, and this trend persists across the different regimes.
- Temperature and specific humidity are not always the most important features, their significance depends on the specific regime.
- The quality of the performance is generally better for the classical network, but when it declines, it does so for both networks.
- Overall, we can conclude that both networks are capable of adapting to different physical conditions, as performance remains good.

4.5 Stability Analysis

In the previous sections, we explored the predictive capabilities of both networks, observed distinct behaviors within each, and identified some differences. Now, we aim to determine whether the observed behaviors are caused by the randomness introduced in different training runs or if they are intrinsic properties of the networks. Specifically, we will assess how stable the results are when training is repeated multiple times, considering that each training run starts from randomly initialized parameters and is influenced by the stochastic nature of the optimization process.
To investigate this, we examine multiple training runs of the same architecture, each yielding a different parameter set due to the randomness in initialization and optimization. Specifically, we selected 10 additional parameter sets for both the classical and quantum networks to evaluate if there are any statistical fluctuations or if the peculiarities observed remain consistent, suggesting that they are not dependent on the particular parameter set. In the following table, we present the performance metrics for all networks to provide an overview.

Network (Classical)	**MSE**	$\mathbf{R^2}$	**Network (Quantum)**	**MSE**	$\mathbf{R^2}$
NN(θ_1)	0.007	0.93	QNN(θ_1)	0.010	0.90
NN(θ_2)	0.006	0.94	QNN(θ_2)	0.009	0.91
NN(θ_3)	0.006	0.94	QNN(θ_3)	0.009	0.90
NN(θ_4)	0.006	0.94	QNN(θ_4)	0.009	0.91
NN(θ_5)	0.006	0.94	QNN(θ_5)	0.011	0.89
NN(θ_6)	0.006	0.94	QNN(θ_6)	0.011	0.89
NN(θ_7)	0.006	0.94	QNN(θ_7)	0.010	0.90
NN(θ_8)	0.006	0.94	QNN(θ_8)	0.010	0.90
NN(θ_9)	0.008	0.92	QNN(θ_9)	0.009	0.91
NN(θ_{10})	0.006	0.94	QNN(θ_{10})	0.010	0.90

Table 9: Summary of MSE and R^2 values for different parameter sets in classical and quantum architectures. The architecture remains consistent across all experiments, while the parameter set θ (labeled with subscripts from 1 to 10) represents the selected set of parameters. All values are computed using a dataset of 100,000 points, as in previous cases.

As shown in the table, both networks maintain relatively stable performance across different training configurations, with only minor variations in MSE and R^2. The classical network consistently outperforms the quantum network.
To quantify the stability of each architecture, we computed the mean and standard deviation of both the MSE and R^2 scores across all 11 training runs (i.e., 10 random initializations plus the one used throughout the thesis). This allows us to estimate the uncertainty associated with the stochastic nature of the training process.

Model	MSE	$\mathbf{R^2}$
Classical NN	0.0063 ± 0.0006	0.937 ± 0.006
Quantum NN	0.0099 ± 0.0008	0.900 ± 0.008

Table 10: Summary of mean and standard deviation for MSE and R^2 across multiple training runs for classical and quantum neural networks.

These results suggest that both architectures produce consistent predictions across multiple training instances. However, the classical network demonstrates slightly lower variance and better average performance.
We now analyze Shapley values to assess feature importance stability across training runs. The bar plots below offer an overview of how feature importance fluctuates across different training runs. Given the high number of analyzed sets, it will be presented an overall bar plot for the classical network and another for the quantum network.

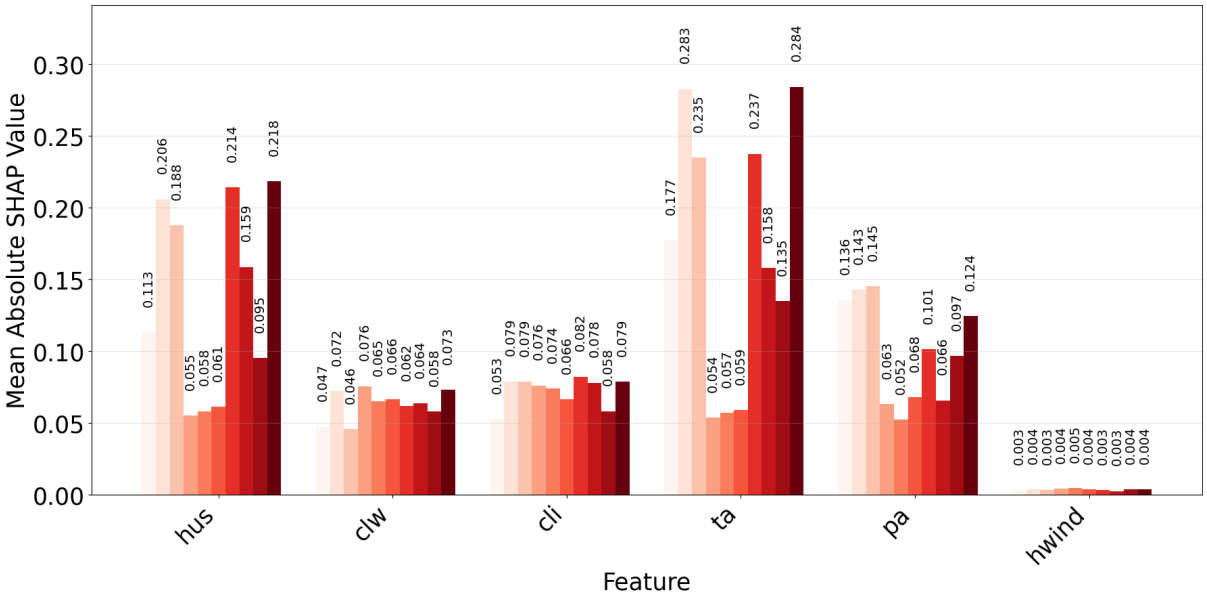

Figure 23: Comparison of feature importance for different parameter sets of classical neural network. The x-axis represents the features, and for each feature, the bars indicate the mean Shapley value across different parameter sets.

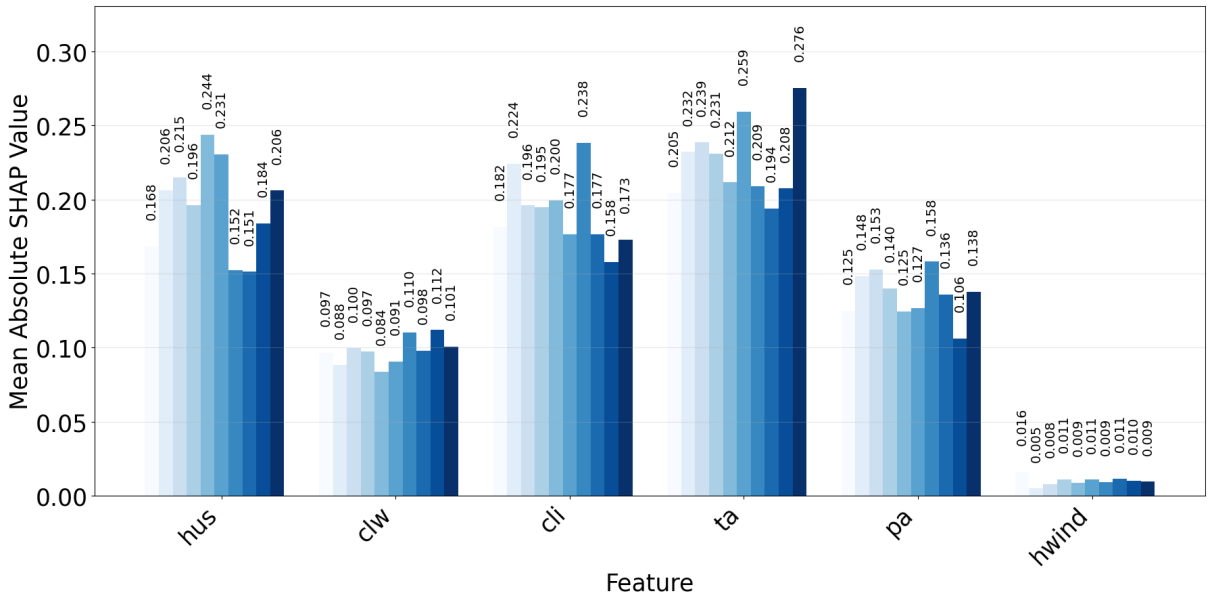

Figure 24: Comparison of feature importance for different optimal parameter sets of the quantum neural network. The x-axis represents the features, and for each feature, the bars indicate the mean Shapley value across different parameter sets.

The quantum network exhibits a more consistent pattern, with clear and compact feature importance distributions, whereas the classical network shows greater variability, particularly for the most influential features (*hus* and *ta*). Some classical parameter sets emphasize these two features, aligning with our previous findings, while others distribute importance more evenly across all features. This variability suggests that some parameter sets, despite minimizing MSE, might not fully capture the key physical processes underlying cloud cover, as identified in our previous analysis.
In contrast, the quantum neural network displays more stable feature importance, with temperature consistently emerging as the most relevant feature, followed by specific humidity and cloud ice at similar levels. Notably, the role of cloud ice remains significant across different parameter sets.
To have a better comparison between the classical and quantum model, a bar plot with all parameter sets from both architectures for each input feature is presented in Figure 25.

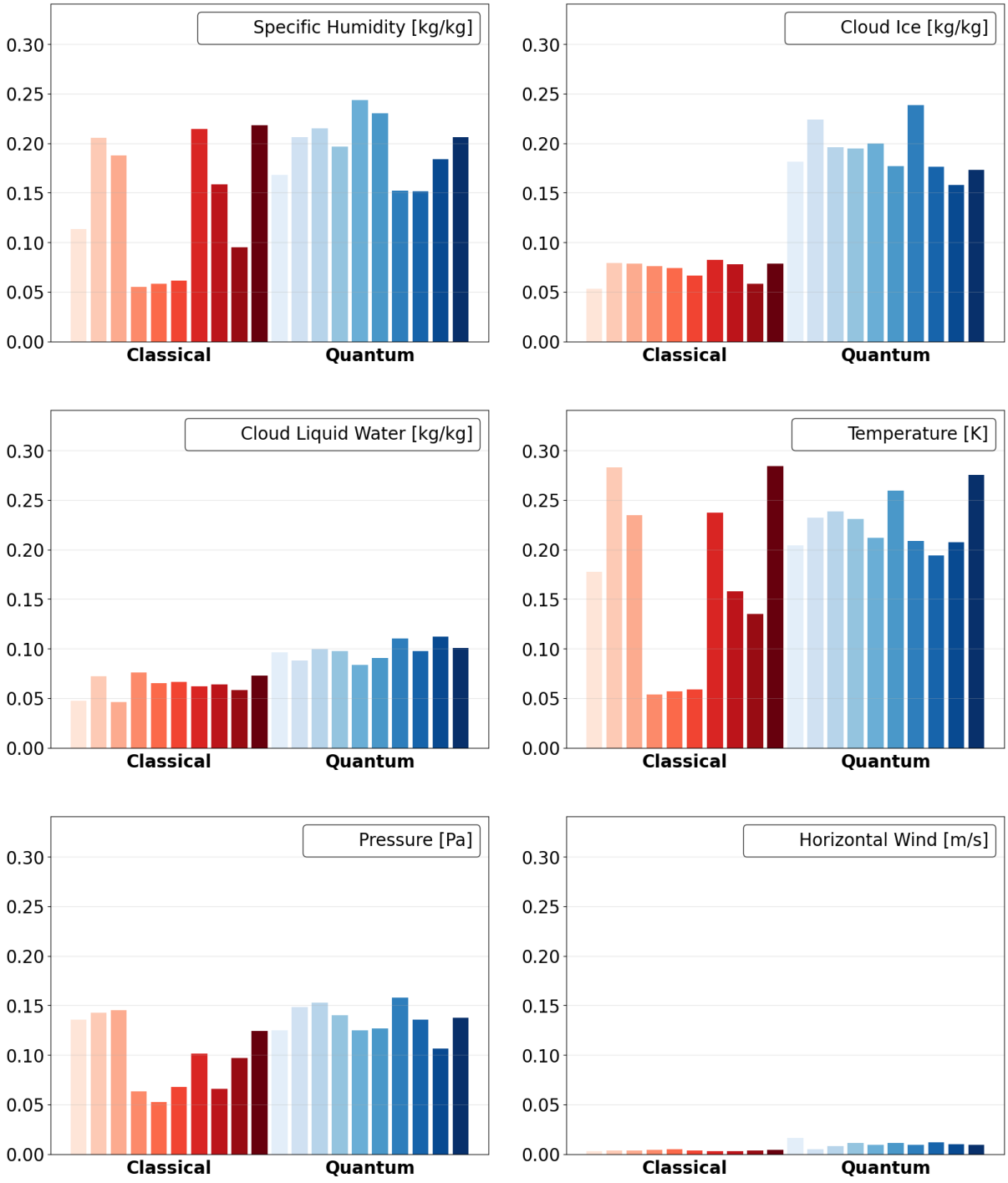

Figure 25: Shapley value distribution for each atmospheric feature across different parameter sets. Each column corresponds to a specific parameter set, and the y-axis represents the Shapley values. The classical neural network is shown in shades of orange, while the quantum neural network is depicted in shades of blue

From these plots, it is evident that the fluctuations in feature importance are more pronounced in the classical architecture compared to its quantum counterpart.
Since we observed variations in the Shapley values for the classical network, we now

investigate whether parameter sets with different mean Shapley values still capture the same physical relationships as those previously analyzed. To do this, we compare the distribution of Shapley values for each feature using two different parameter sets: the one analyzed so far and another from Table 9. Specifically, we select the set $NN^6(\theta_4)$, which exhibits lower mean Shapley values for specific humidity (hus) and temperature (ta) compared to the previously studied set.

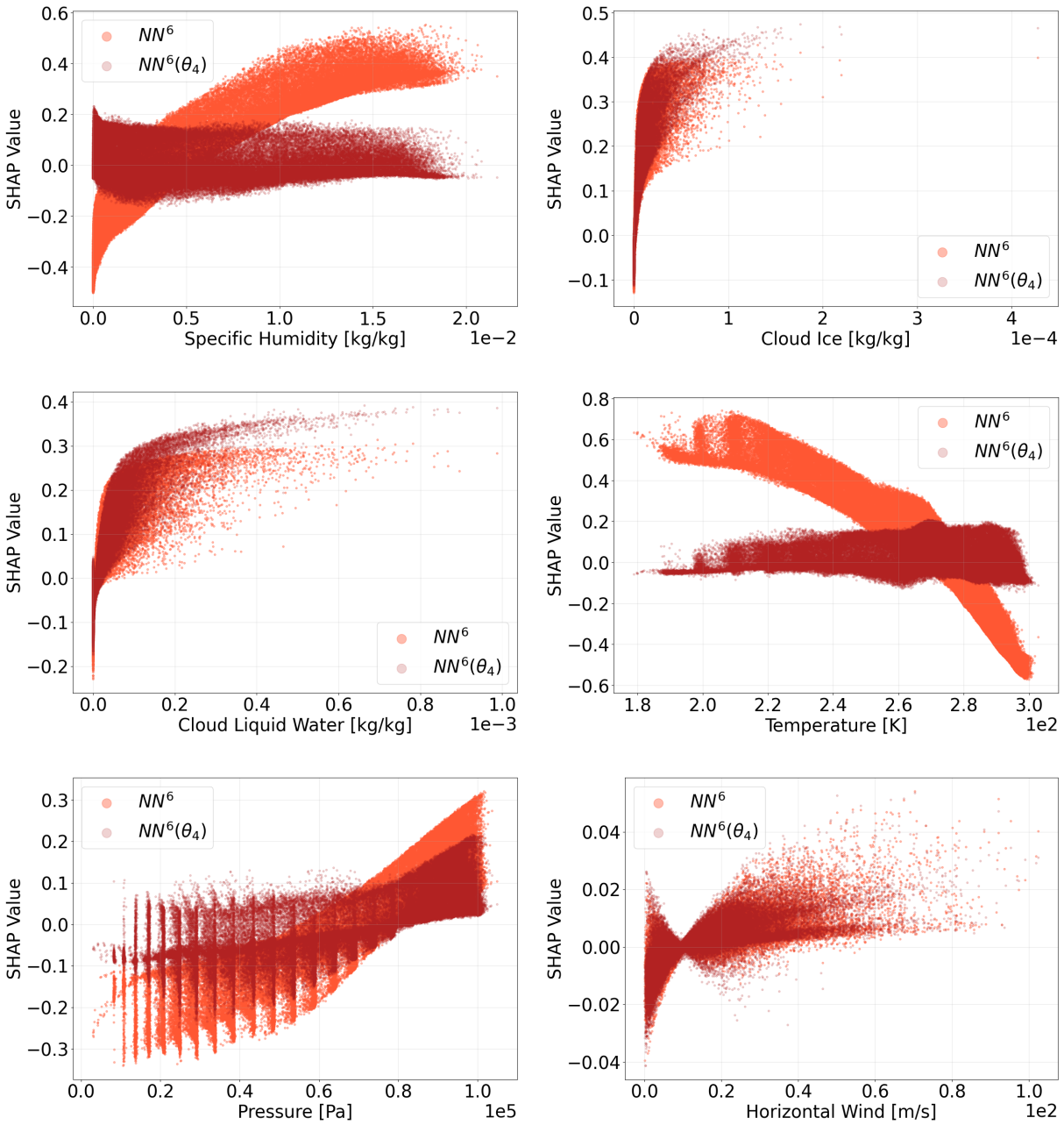

Figure 26: Shapley Value distribution for every atmospheric feature. Each plot contains 100,000 data points. The x-axis represents the feature values, while the y-axis shows the associated SHAP values, indicating the contribution of each feature to the model's predictions. We overlap two different sets of parameters. We compare two parameter sets: the one used in the main study (light red) and another set ($NN^6(\theta_4)$, dark red)

From the plots, we observe that the selected parameter set $NN^6(\theta_4)$ fails to capture certain physical relationships as effectively as the previously analyzed set. Notably, for both specific humidity (hus) and temperature (ta), the Shapley value distributions are nearly flat, suggesting that these features have little to no impact on the model's predictions. This finding highlights that some parameter sets, despite achieving good performance

metrics (e.g., MSE and R^2), may lead to predictions that lack physical meaning.

4.6 Eight Features networks

We now extend the study to networks trained with eight input features instead of six. The goal is to assess whether increasing the number of features enhances the predictive capabilities of the models and whether they continue to learn in the same manner. Additionally, we aim to determine whether the characteristics observed in the 6-feature case persist or if new behaviors emerge. To achieve this, we introduce two additional features: altitude (z_g) and Coriolis force. These features have been selected based on their relative importance among the atmospheric variables [35]. The data preprocessing pipeline remains unchanged from the 6-feature case, ensuring consistency in data handling.
The classical neural network architecture follows a similar structure to the previous model, but with an input layer of size 8 to accommodate the additional features and more hidden layers and parameters. The hidden layers are chosen to keep the total number of trainable parameters comparable to the quantum model. The specific architecture is summarized in Table 11.

NN	Hidden layers	D	Input features
NN^8	$12 \to 6 \to 2$	203	$\{hus, clw, cli, T, pa, h_{wind}, z_g, coriolis\}$

Table 11: Classical neural network architecture with 8 input features. The name NN^8 means that the input layer has 8 neurons, while the hidden layers have 12, 6 and 2 neurons. D stays for the number of parameters. The input features include eight atmospheric variables: specific humidity (hus), cloud liquid water content (clw), cloud ice water content (cli), air temperature (ta), pressure (pa), horizontal wind speed (hwind), altitude (zg) and coriolis force (coriolis).

For the quantum neural network (QNN), we maintain the same ansatz as in the previous case. The quantum model is designed to handle 8 input features while ensuring a fair comparison with its classical counterpart by keeping the number of trainable parameters approximately equal. The details of the quantum architecture are provided in Table 12.

QNN	N	n_{enc}	n_{var}	D	Input features
QNN^8	8	5	3	201	$\{hus, clw, cli, T, pa, h_{wind}, z_g, coriolis\}$

Table 12: Quantum neural network architecture with 8 input features. The architecture is made of $N = 8$ qubits, $n_{enc} = 5$ encoding blocks and $n_{var} = 3$ variational blocks. It has $D = 201$ trainable parameters. The input features include eight atmospheric variables: specific humidity (hus), cloud liquid water content (clw), cloud ice water content (cli), air temperature (ta), pressure (pa), horizontal wind speed (hwind), altitude (zg) and coriolis force (coriolis).

In the following section, we will analyze the performance of these networks and compare their ability to learn and generalize from the extended feature set. To evaluate the

performance of the neural networks, we compare the predictions of both the classical and quantum models against the true values.

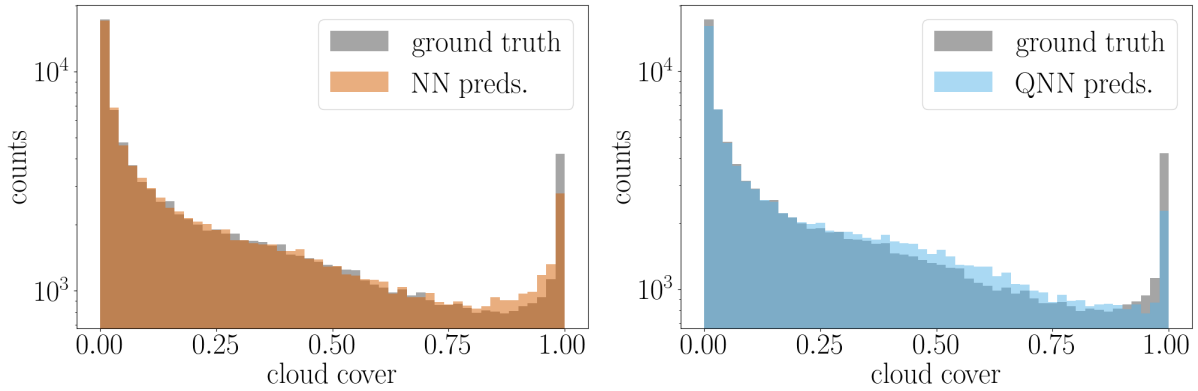

Figure 27: Comparison of histograms: Classical NN (left) and Quantum NN (right). The x-axis represents cloud cover (ranging from 0 to 1), divided into 50 bins, while the y-axis shows frequency counts for each bin, based on a sample of 100,000 test points.

Table 13 summarizes the mean squared error (MSE) and coefficient of determination (R^2) for both models.

Model	**MSE**	R^2
Classical NN (NN^8)	0.008	0.92
Quantum NN (QNN^8)	0.008	0.92

Table 13: Comparison of MSE and R^2 values for Classical and Quantum Neural Networks.

From these results, we observe that in the classical case, adding two extra features does not provide a significant advantage. In the quantum case, however, we see an improvement in performance compared to the 6-feature counterpart. To further investigate the role of the newly introduced features, we analyze feature importance using Shapley values. If these features exhibit high importance, this would indicate that their inclusion has contributed meaningfully to the predictions.
To gain deeper insights into how the networks process the additional features, we now examine feature importance through beeswarm plots, to assess whether the learned feature relationships remain consistent or if new patterns emerge.

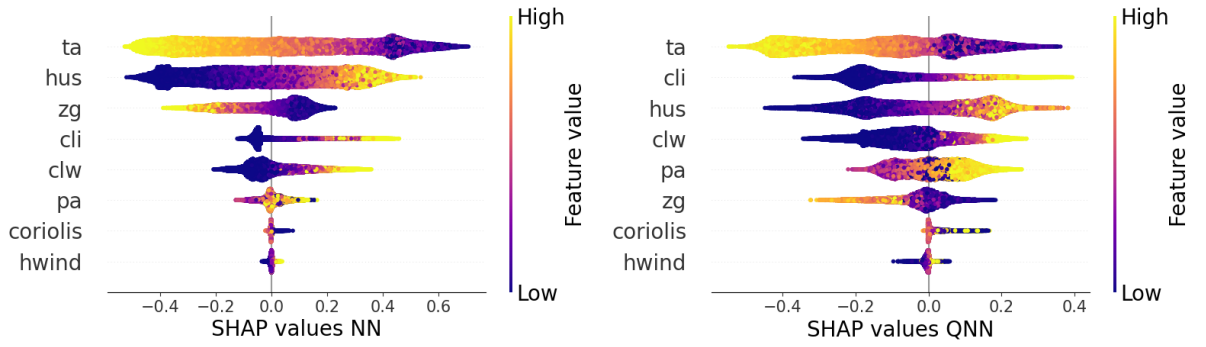

Figure 28: Beeswarm plots for the classical neural network (left) and the quantum neural network (right). Each plot is generated using 100,000 data points. The features are ranked by their average absolute Shapley values, with the most influential features appearing at the top. Each point represents a single data instance, with thicker clusters indicating a higher density of points. The color gradient represents the feature value, ranging from low (purple) to high (yellow).

In this case, temperature (T) remains the most important feature, confirming the trend observed in the 6-feature networks. However, the ranking of importance has changed for some features. For example, the role of ice water content (cli) has become even more pronounced, surpassing specific humidity in importance for the quantum model.
The least important features in this configuration appear to be the Coriolis force and horizontal wind speed (h_{wind}) for both architectures. We can observe that the introduction of additional features does not appear to have significantly impact in the quantum case.
To gain deeper insights, we now examine the Shapley value distributions for each feature, looking for patterns and similarities with the 6-feature case.

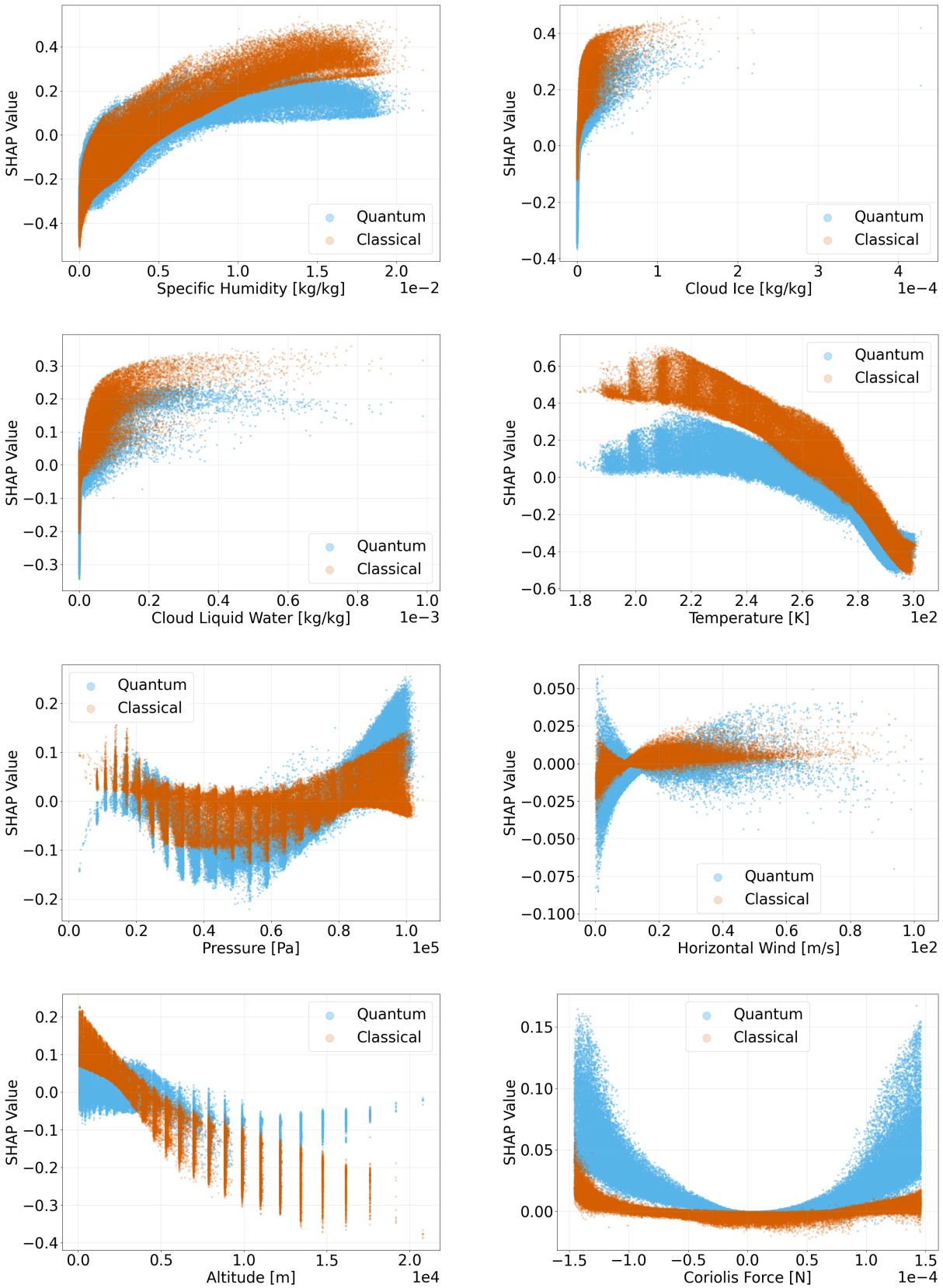

Figure 29: Shapley Values distribution for every atmospheric feature. Each plot contains 100,000 data points. The x-axis represents the feature values, while the y-axis shows the associated Shapley values, indicating the contribution of each feature to the model's predictions. We overlap classical (orange) and quantum (blue).

We observe similar trends as in the previous analysis, with some additional insights:

- **Specific Humidity**: As in the previous case, Shapley values increase with higher specific humidity, indicating its consistent influence on cloud cover predictions.
- **Cloud ice and cloud water**: The same threshold effect is visible: low condensate leads to low cloud cover predictions, while high condensate results in higher Shapley values. Notably, the increased importance of cloud ice is linked to the strong negative Shapley values observed for low cloud ice concentrations.
- **Temperature:** The relationship remains consistent, with lower temperatures favoring higher cloud cover predictions (positive shifts), while higher temperatures result in negative shifts, in line with physical expectations. In the classical case, the distribution appears more skewed.
- **Pressure**: In the classical model, the pressure feature shows a flatter distribution around zero, while in the quantum case, Shapley values initially decrease and then increase.
- **Altitude**: As expected, Shapley values generally decrease with altitude in the classical model. In contrast, in the quantum case, they remain close to zero, suggesting a different sensitivity to altitude in this architecture.
- **Horizontal Wind and Coriolis Force** : Both features have minimal influence on the predictions, particularly in the classical case, reinforcing their lower relevance in determining cloud cover.

5 Conclusion

5.1 Brief Summary of the Research

The main objective of this thesis was to compare classical and quantum neural networks in the context of cloud cover parameterizations, with an emphasis on whether the same physical processes were learned. To evaluate the models, Shapley values were used to assess the importance of each feature and its impact on the prediction. In order to enable a physical interpretation of the results, we also compared the neural networks with a standard parameterization scheme (from Xu and Randall [55]), and analyzed the networks predictions in different cloud regimes. Furthermore, we analyzed the stability of our results against the randomness introduced by the training procedure as well as increasing the number of input features.
In terms of prediction accuracy, the classical network generally outperformed the quantum network, showing slightly higher R^2 values and lower mean squared error (MSE). Regarding feature importance, both networks agreed on the physical influence of specific features on the prediction, showing consistent trends in the Shapley values. At lower temperatures, Shapley values are positive since cooler air holds less moisture, making condensation and cloud formation more likely. Conversely, at higher temperatures, Shapley values turn negative, as warmer air retains more moisture, reducing condensation and cloud cover. imilarly, Shapley values for specific humidity (hus) increase with humidity levels, as higher relative humidity (RH) enhances condensation and cloud formation. For cloud water and cloud ice condensate, higher values of these features correspond to a greater value of cloud cover, as condensation is more likely in the presence of significant amounts of ice and water. Both networks also agreed on which features have minimal impact on the prediction, such as the Coriolis force and horizontal wind speed. However, there were differences in the ranking of features in terms of their importance, for example the quantum network gave overall more importance to cloud ice in comparison to its classical counterpart.
Both networks were able to adapt to different physical regimes without needing to be retrained with data from the specific regime.
Regarding the stability of the results, it was observed that the feature importance in the classical network exhibited more fluctuations across different training runs compared to the quantum network. There were instances where certain sets of parameters in the classical network failed to capture the physically expected relationships between the input variables and the prediction. This suggests that in the classical NN optimization landscape there are functions that, albeit minimizing the loss function (MSE), do not capture the expected physical dependence of the cloud cover on (some of) the input features. This interesting observation underlines the importance of our Shapley analysis in assessing the predictions of a (Q)ML model before its operational use.

5.2 Outlook on future improvements and directions

While this study addresses the initial research questions, it also highlights some limitations and open challenges that pave the way for future investigations.

- First, although the dataset used in this study was large enough for training and testing, it would be interesting to repeat the analysis on different datasets from other storm-resolving simulations, such as NARVAL or QUBICC, to see if the results obtained are consistent.

- To obtain a more comprehensive comparison between classical and quantum models, it would be interesting to analyze several different model architectures to see if performance or insights change, while keeping the number of input features costant.

- The tests were conducted offline, meaning that the neural networks were evaluated independently and not coupled to the climate model. While it is known that good offline performance often correlates with good online performance [94], it is unclear whether the feature importance rankings provided by Shapley values translate into real-time model behavior. This raises the question of whether different parameter sets, despite having similar MSE values, might exhibit distinct physical behaviors when deployed in an online setting.

- Additionally, only one explainable AI method (Shapley values) was used. Future research could explore whether other explainable AI techniques could provide additional insights into the networks beyond feature importance. However, this would require careful consideration of what we aim to learn from the networks in order to choose the most appropriate method.

- The networks analyzed in this study operate on a cell-based approach, meaning that each prediction is made independently for a single atmospheric cell without considering surrounding grid points. A promising extension would be to incorporate information from adjacent cells, at least vertically, to better capture dependencies.

- Another interesting direction would be a comparison with different ML approaches for cloud cover parameterization, such as the equations derived in [30]. This could provide deeper insights into the physical consistency of the learned parameterizations and help to identify which other physical processes the networks might capture.

5.3 Final Considerations

In conclusion, this study contributes to the ongoing effort to improve climate model parameterizations through the use of machine learning models, both classical and quantum, and to the study of the learning capabilities of QNNs in general. We have seen that both networks have their advantages and disadvantages. The classical network excels in terms of performance but sacrifices stability, while the quantum network provides more consistent results, though with slightly lower overall performance.
One important takeaway is that Shapley values, while being a valuable tool for understanding feature importance and the impact of features on predictions, should be seen as a complementary resource to standard metrics such as MSE and R^2. By providing insight into how features influence predictions, Shapley values can help assess whether the learned function aligns with expected physical behaviors, an aspect that is important when deciding which model to deploy in an operational setting. However, since the information

they provide is inherently statistical, Shapley values should be considered as one of several possible tools for evaluating machine learning models, rather than a definitive measure of their reliability or correctness.
Despite these limitations, this study highlights the potential of machine learning and quantum machine learning to enhance parameterizations in climate models. At the same time, our study shows the importance of evaluating such models using XAI methods to infer whether the correct or expected physical behaviors have been captured, before their operational use in climate simulations.

A Appendix A

A.1 Pre-processing

In this appendix we discuss the input and output transformations that we applied to the DYAMOND data after coarse-graining and before feeding them to our classical and quantum models.
For the input features, the idea behind the transformation is to make the feature distribution more uniform within a specified interval, and to still retain the input feature variability in the tails, which can be associated with physical scenarios we are interested in capturing. The transformation function we designed in this case reads as [35]:

$$h(x) = \frac{\log\left(1 + (e-1)\left(\frac{x}{x_{high}}\right)^b\right) - h_0(b, x_{low}, x_{high})}{1 - h_0(b, x_{low}, x_{high})} \tag{33}$$

where $h_0(b, x_{low}, x_{high}) = \log\left(1 + (e-1)(\frac{x_{low}}{x_{high}})^b\right)$ and x_{low}, x_{high} corresponding to the (approximate) minimum and maximum value of the given feature x estimated on the training dataset. We used the following parameters for the input features:

- Specific humidity *hus* [kg/kg] : $b = 0.25$, $x_{low} = 10^{-7}$, $x_{high} = 0.025$,
- Cloud water *clw* [kg/kg] : $b = 0.25$, $x_{low} = 0$, $x_{high} = 0.00145$,
- Cloud ice *cli* [kg/kg] : $b = 0.25$, $x_{low} = 0$, $x_{high} = 0.00055$,
- Horizontal wind *hwind* [m/s] : $b = 0.5$, $x_{low} = 0.0015$, $x_{high} = 115.0$,

The remaining input features are transformed using a simple min-max scaling, and all features (including those listed above) are scaled within the interval $[0, \pi]$ (i.e., we multiplied the above $h(x)$ by a factor π)

A.2 Post-processing

Also the output transformation function $g(x)$ is constructed in order to have the training outputs (targets) in a more uniform distribution compared to the original one in the DYAMOND dataset, which in our case improved the performance of both our quantum and classical models. The transformation function is invertible, and reads as

$$g(x) = \frac{1}{2} + \frac{1}{\pi}\arcsin\left(2\left(\frac{e^{bx^a} - 1}{e^b - 1}\right)^c\right) \tag{34}$$

with parameters $a = 1.29407913$, $b = -3.20011015$, $c = 0.70308237$, which have been chosen in order to have approximate uniformity.

A.3 Shapley Values

The following pseudocode outlines the key steps for computing Shapley Values

```
import shap
K = 100
background_samples = shap.sample(train_inputs, K)
def custom_predict(inputs):
    outs = modelNN.predict(inputs)
    explainer = shap.KernelExplainer(lambda x: custom_predict, background_samples)
shap_values = explainer.shap_values(test_inputs)
shap_explanation = shap.Explanation(
    values=shap_values,
    base_values=expected_value,
    data=test_inputs,
    feature_names=features_kept)
```

1. **Sampling the Background Data**
   Since Kernel SHAP requires a reference dataset to estimate Shapley values, we randomly sample $K = 100$ data points from the training set (`train_inputs`). This serves as the background distribution.

2. **Defining the Custom Prediction Function**
   A function `custom_predict` wraps the model's prediction method (that in our case can be the classical or quantum model). This function is used to evaluate the effect of different feature subsets during the SHAP computation.

3. **Initializing the Kernel SHAP Explainer**
   The `shap.KernelExplainer` is instantiated using the `custom_predict` function and the background samples.

4. **Computing Shapley Values**
   The `shap_values` are computed for the test dataset (`test_inputs`), quantifying the importance of each feature for the model's output.

5. **Creating a SHAP Explanation Object**
   The results are stored in a `shap.Explanation` object, which contains:

   - `values`: The computed Shapley values.
   - `base_values`: The expected value of the model's output.
   - `data`: The input data.
   - `feature_names`: Names of the input features.

References

[1] Intergovernmental Panel on Climate Change (IPCC). Climate change 2021: The physical science basis. Contribution of working group i to the sixth assessment report of the intergovernmental panel on climate change, IPCC, 2021.

[2] Andrew Gettelman and Richard B. Rood. *Demystifying Climate Models: A Users Guide to Earth System Models*, volume 2 of *Earth Systems Data and Models*. Springer, 2016.

[3] P. N. Edwards. History of climate modelling. *Wiley Interdisciplinary Reviews: Climate Change*, 2010.

[4] Manuel Schlund, Veronika Eyring, Gustau Camps-Valls, Pierre Friedlingstein, Pierre Gentine, and Markus Reichstein. Constraining uncertainty in projected gross primary production with machine learning. *Journal of Geophysical Research: Biogeosciences*, 125(11):e2019JG005619, 2020.

[5] Gustau Camps-Valls, Devis Tuia, Xiao Xiang Zhu, and Markus Reichstein. *Deep Learning for the Parametrization of Subgrid Processes in Climate Models*, chapter 21, pages 307–314. Wiley, 2021.

[6] M. A. Giorgetta, R. Brokopf, T. Crueger, M. Esch, S. Fiedler, J. Helmert, C. Hohenegger, L. Kornblueh, M. Köhler, E. Manzini, T. Mauritsen, C. Nam, T. Raddatz, S. Rast, and D. Reinert. ICON-A, the Atmosphere Component of the ICON Earth System Model: I. Model Description. *Journal of Advances in Modeling Earth Systems*, 10, 2018.

[7] N. McFarlane. Parameterizations: Representing key processes in climate models without resolving them. *Wiley Interdisciplinary Reviews: Climate Change*, 2011.

[8] D. J. Stensrud. *Parameterization Schemes: Keys to Understanding Numerical Weather Prediction Models*. Cambridge University Press, 2007.

[9] V. Eyring, V. Mishra, G. P. Griffith, L. Chen, T. Keenan, M. R. Turetsky, S. Brown, F. Jotzo, F. C. Moore, and S. van der Linden. Reflections and projections on a decade of climate science. *Nature Climate Change*, 11:279–285, 2021.

[10] C. Tebaldi, K. Debeire, V. Eyring, E. Fischer, J. Fyfe, P. Friedlingstein, R. Knutti, J. Lowe, B. O'Neill, B. Sanderson, D. van Vuuren, K. Riahi, M. Meinshausen, Z. Nicholls, K. B. Tokarska, G. Hurtt, E. Kriegler, J.-F. Lamarque, G. Meehl, R. Moss, S. E. Bauer, O. Boucher, V. Brovkin, Y.-H. Byun, M. Dix, S. Gualdi, H. Guo, J. G. John, S. Kharin, Y. Kim, T. Koshiro, L. Ma, D. Olivié, S. Panickal, F. Qiao, X. Rong, N. Rosenbloom, M. Schupfner, R. Séférian, A. Sellar, T. Semmler, X. Shi, Z. Song, C. Steger, R. Stouffer, N. Swart, K. Tachiiri, Q. Tang, H. Tatebe, A. Voldoire, E. Volodin, K. Wyser, X. Xin, S. Yang, Y. Yu, and T. Ziehn. Climate model projections from the Scenario Model Intercomparison Project (ScenarioMIP) of CMIP6. *Earth System Dynamics*, 12(1):253–293, 2021.

[11] Veronika Eyring, Sandrine Bony, Gerald Meehl, Catherine Senior, Bjorn Stevens, Stouffer Ronald, and Karl Taylor. Overview of the coupled model intercomparison project phase 6 (cmip6) experimental design and organization. *Geoscientific Model Development*, 9:1937–1958, 05 2016.

[12] U. Lohmann, F. Lüönd, and F. Mahrt. *An introduction to clouds: From the microscale to climate*. Cambridge University Press, 2016.

[13] Markus Quante. The role of clouds in the climate system. *Journal De Physique Iv - J PHYS IV*, 121:61–86, 12 2004.

[14] Hannah Christensen and Laure Zanna. Parametrization in weather and climate models, 12 2022.

[15] Veronika Eyring, Pierre Gentine, Gustau Camps-Valls, David M. Lawrence, and Markus Reichstein. Ai-empowered next-generation multiscale climate modelling for mitigation and adaptation. *Nature Geoscience*, 17:963–971, 2024.

[16] Noah D. Brenowitz, Brian Henn, Jeremy McGibbon, Spencer K. Clark, Anna Kwa, W. Andre Perkins, Oliver Watt-Meyer, and Christopher S. Bretherton. Machine learning climate model dynamics: Offline versus online performance, 2020.

[17] Veronika Eyring, William D. Collins, Pierre Gentine, Elizabeth A. Barnes, Marcelo Barreiro, Tom Beucler, Marc Bocquet, Christopher S. Bretherton, Hannah M. Christensen, Katherine Dagon, David John Gagne, David Hall, Dorit Hammerling, Stephan Hoyer, Fernando Iglesias-Suarez, Ignacio Lopez-Gomez, Marie C. McGraw, Gerald A. Meehl, Maria J. Molina, Claire Monteleoni, Juliane Mueller, Michael S. Pritchard, David Rolnick, Jakob Runge, and Laure Zanna. Pushing the frontiers in climate modelling and analysis with machine learning. *Nature Climate Change*, 14:916–928, 2024.

[18] F. Chevallier, F. Chéruy, N. Scott, and A. Chédin. A neural network approach for a fast and accurate computation of a longwave radiative budget. *Journal of Applied Meteorology*, 37(11):1385–1397, 1998.

[19] Vladimir M. Krasnopolsky, Michael S. Fox-Rabinovitz, and Dmitry V. Chalikov. New approach to calculation of atmospheric model physics: Accurate and fast neural network emulation of longwave radiation in a climate model. *Monthly Weather Review*, 133:1370–1383, 2005.

[20] Ryan Lagerquist, David Turner, Imme Ebert-Uphoff, Jebb Stewart, and Venita Hagerty. Using deep learning to emulate and accelerate a radiative transfer model. *Journal of Atmospheric and Oceanic Technology*, 38:1673–1696, 2021. Online Publication: 21 Sep 2021, Print Publication: 01 Oct 2021.

[21] Katharina Hafner, Fernando Iglesias-Suarez, Sara Shamekh, and et al. Interpretable machine learning-based radiation emulation for icon. *ESS Open Archive*, 11 2024.

[22] Vladimir M. Krasnopolsky. *The Application of Neural Networks in the Earth System Sciences: Neural Networks Emulations for Complex Multidimensional Mappings*, volume 1 of *Atmospheric and Oceanographic Sciences Library*. Springer Dordrecht, 2013.

[23] S. Rasp, M. S. Pritchard, and P. Gentine. Deep learning to represent subgrid processes in climate models. *PNAS*, 115(39):9684–9689, 2018.

[24] P. Gentine, M. Pritchard, S. Rasp, G. Reinaudi, and G. Yacalis. Could machine learning break the convection parameterization deadlock? *Geophysical Research Letters*, 45(11):5742–5751, 5 2018.

[25] N. D. Brenowitz and C. S. Bretherton. Spatially extended tests of a neural network parametrization trained by coarse-graining. *Journal of Advances in Modeling Earth Systems*, 11(8):2728–2744, 2019.

[26] Janni Yuval and Paul A. O'Gorman. Stable machine-learning parameterization of subgrid processes for climate modeling at a range of resolutions. *Nature Communications*, 11, 2020.

[27] Helge Heuer, Mierk Schwabe, Pierre Gentine, Marco A. Giorgetta, and Veronika Eyring. Interpretable multiscale machine learning-based parameterizations of convection for icon. *Journal of Advances in Modeling Earth Systems*, 2024.

[28] A. Grundner, T. Beucler, P. Gentine, F. Iglesias-Suarez, M. A. Giorgetta, and V. Eyring. Deep Learning Based Cloud Cover Parameterization for ICON. *Journal of Advances in Modeling Earth Systems (JAMES)*, 14, 2022.

[29] Arthur Grundner. *Data-Driven Cloud Cover Parameterizations for the ICON Earth System Model Using Deep Learning and Symbolic Regression*. PhD thesis, Bremen University, 2023.

[30] A. Grundner, T. Beucler, P. Gentine, and V. Eyring. Data-driven equation discovery of a cloud cover parameterization. *Journal of Advances in Modeling Earth Systems (JAMES)*, 16, 2024.

[31] M. Cerezo, G. Verdon, H.-Y. Huang, L. Cincio, and P. J. Coles. Challenges and opportunities in quantum machine learning. *Nature Computational Science*, 2:567–576, 2022.

[32] A. Abbas, D. Sutter, C. Zoufal, A. Lucchi, A. Figalli, and S. Woerner. The power of quantum neural networks. *Nature Computational Science*, 1:403–409, 2021.

[33] Alejandro Perdomo-Ortiz, Marcello Benedetti, John Realpe-Gómez, and Rupak Biswas. Opportunities and challenges for quantum-assisted machine learning in near-term quantum computers. *Quantum Science and Technology*, 3(3):030502, June 2018.

[34] F. Tennie and T. N. Palmer. Quantum computers for weather and climate prediction: The good, the bad, and the noisy. *Bulletin of the American Meteorological Society*, 104:E488, 2023.

[35] Lorenzo Pastori, Arthur Grundner, Veronika Eyring, and Mierk Schwabe. Quantum neural networks for cloud cover parameterizations in climate models, 2025.

[36] Mierk Schwabe, Lorenzo Pastori, Inés de Vega, Pierre Gentine, Luigi Iapichino, Valtteri Lahtinen, Martin Leib, Jeanette M. Lorenz, and Veronika Eyring. Opportunities and challenges of quantum computing for climate modelling, 2025.

[37] L. Sünkel, D. Martyniuk, J. J. Reichwald, A. Morariu, R. H. Seggoju, P. Altmann, C. Roch, and A. Paschke. Hybrid quantum machine learning assisted classification of covid-19 from computed tomography scans. In *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pages 356–366. IEEE, 2023.

[38] T. Hur, L. Kim, and D. K. Park. Quantum convolutional neural network for classical data classification. *Quantum Machine Intelligence*, 4:3, 2022.

[39] Joseph Bowles, Shahnawaz Ahmed, and Maria Schuld. Better than classical? the subtle art of benchmarking quantum machine learning models, 2024.

[40] M. Ancona, E. Ceolini, C. Öztireli, and M. Gross. Gradient-based attribution methods. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, page 169–191. Springer, 2019.

[41] Feiyu Xu, Hans Uszkoreit, Yangzhou Du, Wei Fan, Dongyan Zhao, and Jun Zhu. Explainable ai: A brief survey on history, research areas, approaches and challenges. In Jie Tang, Min-Yen Kan, Dongyan Zhao, Sujian Li, and Hongying Zan, editors, *Natural Language Processing and Chinese Computing*, pages 563–574, Cham, 2019. Springer International Publishing.

[42] Christoph Molnar. *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*. Springer, 2020.

[43] Andreas Holzinger, Randy Goebel, Ruth Fong, Taesup Moon, Klaus-Robert Müller, and Wojciech Samek. *xxAI - Beyond Explainable AI*. Springer, 2022.

[44] Patrick Steinmüller and Tobias Schulz and Ferdinand Graf and Daniel Herr. explainable ai for quantum machine learning, 2022.

[45] Benedek Rozemberczki, Lauren Watson, Péter Bayer, Hao-Tsung Yang, Olivér Kiss, Sebastian Nilsson, and Rik Sarkar. The shapley value in machine learning, 2022.

[46] I. E. Kumar, S. Venkatasubramanian, C. Scheidegger, and S. Friedler. Problems with Shapley-value-based explanations as feature importance measures. In *International Conference on Machine Learning*, pages 5491–5500, 2020.

[47] Luke Power and Krishnendu Guha. Feature importance and explainability in quantum machine learning, 2024.

[48] Klim-QML Project. Klim-qml: Quantum machine learning for climate models, 2025.

[49] World Meteorological Organization. WMO Statement on the State of the Global Climate in 2020. `https://public.wmo.int/en/our-mandate/climate/wmo-statement-state-global-climate`, 2021. Report No. 1289.

[50] NASA. About earth system: Background information. `https://mynasadata.larc.nasa.gov/basic-page/about-earth-system-background-information`, 2024.

[51] T. Schneider, J. Teixeira, C. S. Bretherton, F. Brient, K. G. Pressel, C. Schar, and A. P. Siebesma. Climate goals and computing the future of clouds. *Nature Climate Change*, 7(1):3–5, 2017.

[52] Omid Alizadeh. Advances and challenges in climate modeling. *Springer Nature B.V.*, 2022.

[53] Adrian M. Tompkins. Cloud parametrization. *Abdus Salam International Centre for Theoretical Physics (ICTP)*, 2008.

[54] H. Sundqvist, E. Berge, and J. E. Kristjánsson. Condensation and cloud parameterization studies with a mesoscale numerical weather prediction model. *Monthly Weather Review*, 1989.

[55] K.-M. Xu and D. A. Randall. A semiempirical cloudiness parameterization for use in climate models. *Journal of the Atmospheric Sciences*, 53(21):3084–3102, 1996.

[56] Y. Wang, S. Yang, G. Chen, Q. Bao, and J. Li. Evaluating two diagnostic schemes of cloud-fraction parameterization using the cloudsat data. *Atmospheric Research*, 282, 2023.

[57] C. J. Walcek. Cloud cover and its relationship to relative humidity during a springtime midlatitude cyclone. *Monthly Weather Review*, 122(6):1021–1035, 1994.

[58] F. Prill, D. Reinert, D. Rieger, and G. Zängl. *Working with the ICON Model*, 2024.

[59] F. Prill, D. Reinert, D. Rieger, and G. Zängl. *Working with the ICON Model*. Max-Planck-Institut für Meteorologie, 6 2024.

[60] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. Deep learning for computer vision: A brief review. *Computational Intelligence and Neuroscience*, 2018:7068349, 2018.

[61] Erwin Loh. Medicine and the rise of the robots: a qualitative review of recent advances of artificial intelligence in health. *BMJ Leader*, 2(2):59–63, 2018.

[62] Matthew Chantry, Sam Hatfield, Peter Dueben, Inna Polichtchouk, and Tim Palmer. Machine learning emulation of gravity wave drag in numerical weather forecasting. *Journal of Advances in Modeling Earth Systems*, 13(6), 2021.

[63] Hervé Abdi, Dominique Valentin, and Berthe Edelman. *Neural Networks*. SAGE Publications, 1999.

[64] Terrence L. Fine. *Feedforward Neural Network Methodology*. Springer, 2006.

[65] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[66] M. C. Caro, H.-Y. Huang, M. Cerezo, K. Sharma, A. Sornborger, L. Cincio, and P. J. Coles. Generalization in quantum machine learning from few training data. *Nature Communications*, 13, 2022.

[67] T. Haug and M. S. Kim. Generalization of Quantum Machine Learning Models Using Quantum Fisher Information Metric. *Physical Review Letters*, 133:050603, 2024.

[68] Kyriaki A. Tychola, Theofanis Kalampokas, and George A. Papakostas. Quantum machine learning—an overview. *Electronics*, 12(11):2379, 2023.

[69] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. Quantum machine learning. *Nature*, 549:195–202, 2017.

[70] Amal Nammouchi, Andreas Kassler, and Andreas Theorachis. Quantum Machine Learning in Climate Change and Sustainability: a Review, 2023.

[71] A. Sakhnenko, J. Sikora, and J. Lorenz. Building continuous quantum-classical bayesian neural networks for a classical clinical dataset. In *Proceedings of Recent Advances in Quantum Computing and Technology, ReAQCT '24*, pages 62–72. ACM, 2024.

[72] S. S. Lachure, A. Lohidasan, A. Tiwari, M. Dhabu, and N. D. Bokde. Quantum machine learning applications to address climate change: A short review. In *Handbook of Research on Quantum Computing for Smart Environments*, chapter 4. IGI Global Scientific Publishing, 2023.

[73] Mukta Nivelkar, Sunil Bhirud, Manmeet Singh, Rahul Ranjan, and Bipin Kumar. Quantum computing to study cloud turbulence properties. *IEEE Access*, 11:70679–70690, 2023.

[74] Soronzonbold Otgonbaatar and Dieter Kranzlmüller. Quantum-inspired tensor network for Earth science, 2023.

[75] Jonathan Wei Zhong Lau, Kian Hwee Lim, Harshank Shrotriya, and Leong Chuan Kwek. NISQ computing: where are we and where do we go? *AAPPS Bulletin*, 32(2):27, 2022.

[76] M. Cerezo, Martin Larocca, Diego García-Martín, N. L. Diaz, Paolo Braccia, Enrico Fontana, Manuel S. Rudolph, Pablo Bermejo, Aroosa Ijaz, Supanut Thanasilp, Eric R. Anschuetz, and Zoë Holmes. Does provable absence of barren plateaus imply classical simulability? or, why we need to rethink variational quantum computing, 2024.

[77] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, 2000.

[78] M. Schuld, R. Sweke, and J. J. Meyer. Effect of data encoding on the expressive power of variational quantum machine-learning models. *Physical Review A*, 103:032430, 2021.

[79] M. C. Caro, E. Gil-Fuster, J. J. Meyer, J. Eisert, and R. Sweke. Encoding-dependent generalization bounds for parametrized quantum circuits. *Quantum*, 5:582, 2021.

[80] Pierre Gentine and Veronika Eyring. Better better — ml for improved climate models and projections — ai for good. `https://www.youtube.com/watch?app=desktop&v=SYC7ZVLxomI&t=336s`, 2024.

[81] D. Silver. Mastering the game of go without human knowledge. *Nature*, 550:354–359, 2017.

[82] S. Dieleman, K. Willett, and J. Dambre. Rotation-invariant convolutional neural networks for galaxy morphology prediction. *Mon. Not. Roy. Astron. Soc.*, 450:1441–1459, 2015.

[83] Alexander Rakhlin, Alexey Shvets, Vladimir Iglovikov, and Alexandr A. Kalinin. *Deep Convolutional Neural Networks for Breast Cancer Histology Image Analysis*, page 737–744. Springer International Publishing, 2018.

[84] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning, 2017.

[85] Center for Data Innovation. The EU should clarify the distinction between explainability and interpretability in the AI act, 2022.

[86] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 4765–4774, 2017. Paul G. Allen School of Computer Science, University of Washington.

[87] Lloyd Shapley. *A Value for N-Person Games*, volume 2. Princeton University Press, Princeton, NJ, 1953.

[88] R. Heese, T. Gerlach, S. Mücke, S. Müller, M. Jakobs, and N. Piatkowski. Explaining Quantum Circuits with Shapley Values: Towards Explainable Quantum Machine Learning. *Quantum Machine Intelligence*, 2025.

[89] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: Large-scale machine learning on heterogeneous distributed systems, 2016.

[90] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii. Data re-uploading for a universal quantum classifier. *Quantum*, 2:100, 2018.

[91] Ville Bergholm, Josh Izaac, Maria Schuld, Christian Gogolin, Shahnawaz Ahmed, Vishnu Ajith, M. Sohaib Alam, Guillermo Alonso-Linaje, B. AkashNarayanan, Ali Asadi, Juan Miguel Arrazola, Utkarsh Azad, Sam Banning, Carsten Blank, Thomas R Bromley, Benjamin A. Cordier, Jack Ceroni, Alain Delgado, Olivia Di Matteo, Amintor Dusko, Tanya Garg, Diego Guala, Anthony Hayes, Ryan Hill, Aroosa Ijaz, Theodor Isacsson, David Ittah, Soran Jahangiri, Prateek Jain, Edward Jiang, Ankit Khandelwal, Korbinian Kottmann, Robert A. Lang, Christina Lee, Thomas Loke, Angus Lowe, Keri McKiernan, Johannes Jakob Meyer, J. A. Montañez-Barrera, Romain Moyard, Zeyue Niu, Lee James O'Riordan, Steven Oud, Ashish Panigrahi, Chae-Yeun Park, Daniel Polatajko, Nicolás Quesada, Chase Roberts, Nahum Sá, Isidor Schoch, Borun Shi, Shuli Shu, Sukin Sim, Arshpreet Singh, Ingrid Strandberg, Jay Soni, Antal Száva, Slimane Thabet, Rodrigo A. Vargas-Hernández, Trevor Vincent, Nicola Vitucci, Maurice Weber, David Wierichs, Roeland Wiersema, Moritz Willmann, Vincent Wong, Shaoming Zhang, and Nathan Killoran. Pennylane: Automatic differentiation of hybrid quantum-classical computations, 2022.

[92] Bjorn Stevens, Masaki Satoh, Ludovic Auger, Joachim Biercamp, Christopher S. Bretherton, Xi Chen, Peter Düben, Falko Judt, Marat Khairoutdinov, Daniel Klocke, Chihiro Kodama, Luis Kornblueh, Shian-Jiann Lin, Philipp Neumann, William M. Putman, Niklas Röber, Ryosuke Shibuya, Benoit Vanniere, Pier Luigi Vidale, Nils Wedi, and Linjiong Zhou. DYAMOND: the DYnamics of the Atmospheric general circulation Modeled On Non-hydrostatic Domains. *Progress in Earth and Planetary Science*, 6(61), 2019.

[93] Julia Duras, Florian Ziemen, and Daniel Klocke. The DYAMOND Winter data collection. Presentation at EGU General Assembly 2021, session AS4.2, 2021.

[94] D. J. Gagne, H. M. Christensen, A. C. Subramanian, and A. H. Monahan. Machine learning for stochastic parameterization: Generative adversarial networks in the Lorenz'96 model. *Journal of Advances in Modeling Earth Systems*, 12(3):e2019MS001896, 2020.

[95] L. Pira and C. Ferrie. On the interpretability of quantum neural networks. *Quantum Machine Intelligence*, 6(2), 2024.

[96] P. D. Dueben and P. Bauer. Challenges and design choices for global weather and climate models based on machine learning. *Geoscientific Model Development*, 11(10):3999–4009, 2018.

[97] M. Tiedtke. A semiempirical cloudiness parameterization for use in climate models. *Journal of the Atmospheric Sciences*, 53(21), 1996.

[98] Ali Ramadhan, John Marshall, Andre Souza, Xin Kai Lee, Ulyana Piterbarg, Adeline Hillier, Gregory LeClaire Wagner, Christopher Rackauckas, Chris Hill, Jean-Michel Campin, and Raffaele Ferrari. Capturing missing physics in climate model parameterizations using neural differential equations, 2023.

[99] Mukund Sundararajan and Amir Najmi. The many Shapley values for model explanation, 2020.

[100] Mark Z. Jacobson. *Fundamentals of Atmospheric Modeling*. Cambridge University Press, 2 edition, 2005.

[101] Marcello Benedetti, Erika Lloyd, Stefan Sack, and Mattia Fiorentini. Parameterized quantum circuits as machine learning models. *Quantum Science and Technology*, 4(4):043001, November 2019.

[102] Y. Du, M.-H. Hsieh, T. Liu, and D. Tao. Expressive power of parametrized quantum circuits. *Physical Review Research*, 2:033125, 2020.

[103] Franz J. Schreiber, Jens Eisert, and Johannes Jakob Meyer. Classical surrogates for quantum learning models. *Physical Review Letters*, 131(10), 2023.

[104] Jonas Landman, Slimane Thabet, Constantin Dalyac, Hela Mhiri, and Elham Kashefi. Classically approximating variational quantum machine learning with random fourier features, 2022.

[105] G. M. Flato. Earth system models: An overview. *Wiley Interdisciplinary Reviews: Climate Change*, 2011.

[106] C. Irrgang, N. Boers, and M. et al. Sonnewald. Towards Neural Earth System Modelling by Integrating Artificial Intelligence in Earth System Science. *Nature Machine Intelligence*, 3:667–674, 2021.

[107] C. Hill, C. DeLuca, Balaji, M. Suarez, and A. Da Silva. The Architecture of the Earth System Modeling Framework. *Computing in Science & Engineering*, 6(1):18–28, 2004.

[108] T. Beucler, I. Ebert-Uphoff, S. Rasp, M. Pritchard, and P. Gentine. Machine learning for clouds and climate. In S. C. Sullivan and C. Hoose, editors, *Clouds and their climate impact*. John Wiley & Sons, 2023.

[109] K. Kashinath, M. Mustafa, A. Albert, et al. Physics-informed machine learning: case studies for weather and climate modelling. *Philosophical Transactions of the Royal Society A*, 379:20200093, 2021.

[110] H. W. Barker, G. Stephens, P. Partain, et al. Assessing 1d atmospheric solar radiative transfer models: Interpretation and handling of unresolved clouds. *Journal of Climate*, 16(16):2676–2699, 2003.

[111] T. G. Beucler, I. Ebert-Uphoff, S. Rasp, M. Pritchard, and P. Gentine. Machine learning for clouds and climate. *Earth Space Sci. Open Arch.*, 2020.

[112] A. Gettelman, D. J. Gagne, C.-C. Chen, et al. Machine learning the warm rain process. *Journal of Advances in Modeling Earth Systems*, 13(2), 2021.

[113] João Teixeira. Cloud fraction and relative humidity in a prognostic cloud fraction scheme. *Quarterly Journal of the Royal Meteorological Society*, 127(577):2045–2064, 2001.

[114] H. Sundqvist. A parameterization scheme for non-convective condensation including prediction of cloud water content. *Quarterly Journal of the Royal Meteorological Society*, 104(441):677–690, 1978.

[115] S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

[116] J. Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

[117] M. Satoh, B. Stevens, F. Judt, M. Khairoutdinov, S.-J. Lin, W. M. Putman, and P. Düben. Global cloud-resolving models. *Current Climate Change Reports*, 5:172–184, 2019.

[118] W. Samek, G. Montavon, A. Vedaldi, L. K. Hansen, and K.-R. Müller. *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, volume 11700. Springer Nature, 2019.

[119] P. A. O'Gorman and J. G. Dwyer. Using machine learning to parameterize moist convection: Potential for modeling of climate, climate change, and extreme events. *Journal of Advances in Modeling Earth Systems*, 10(10):2548–2563, 2018.

[120] A. Kwa, S. K. Clark, B. Henn, N. D. Brenowitz, J. McGibbon, O. Watt-Meyer, W. A. Perkins, L. Harris, and C. S. Bretherton. Machine-learned climate model corrections from a global storm-resolving model: Performance across the annual cycle. *Journal of Advances in Modeling Earth Systems*, 15(5):e2022MS003400, 2023.

[121] S. Y.-C. Chen, T.-C. Wei, C. Zhang, H. Yu, and S. Yoo. Quantum convolutional neural networks for high energy physics data analysis. *Physical Review Research*, 4:013231, 2022.

[122] S. M. Rahman, O. H. Alkhalaf, M. S. Alam, S. P. Tiwari, M. Shafiullah, S. M. Al-Judaibi, and F. S. Al-Ismail. Climate change through quantum lens: Computing and machine learning. *Earth Systems and Environment*, 8:705–718, 2024.

[123] Amy McGovern, Ryan Lagerquist, David John Gagne, G. Eli Jergensen, Kimberly L. Elmore, Cameron R. Homeyer, and Travis Smith. Making the black box more transparent: Understanding the physical implications of machine learning. *The 1st NOAA Workshop on Leveraging AI in the Exploitation of Satellite Earth Observations & Numerical Weather Prediction*, pages 2175–2199, 2019. Print Publication: 01 Nov 2019.

[124] Plamen P. Angelov, Eduardo A. Soares, Richard Jiang, Nicholas I. Arnold, and Peter M. Atkinson. Explainable artificial intelligence: an analytical review. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 11(4):e1424, 2021. First published: 12 July 2021.

[125] Leilani H. Gilpin, David Bau, Ben Z. Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. `https://arxiv.org/abs/1806.00069`, 2018.

[126] Erik Strumbelj and Igor Kononenko. An efficient explanation of individual classifications using game theory. *KDD 2010 Workshop on Knowledge Discovery and Data Mining*, pages 1–4, 2010.

[127] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

[128] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Dino Pedreschi, and Fosca Giannotti. A survey of methods for explaining black box models. *ACM Computing Surveys*, 51(5):93:1–93:42, 2018.

Declaration of authorship

I hereby declare that this thesis is my own work, and that I have not used any sources and aids other than those stated in the thesis.

Location, date

Name

Acknowledgements

I would like to thank the DYAMOND team for sharing their data, which was essential for this study.
This project was made possible by the DLR Quantum Computing Initiative and the Federal Ministry for Economic Affairs and Climate Action; qci.dlr.de/projects/klim-qml
I am grateful to Mierk Schwabe for offering me the opportunity to work on this project.
I would also like to sincerely thank my supervisor, Jeanette Lorenz, for her valuable support and advice along the way.
A special thanks goes to Lorenzo Pastori for his guidance, mentorship, and for everything he taught me during this journey.
I am grateful as well to the KLIM-QML team and the members of the Institute of Atmospheric Physics for warmly welcoming me and making this experience so enriching.
Finally, I would like to thank my family and friends for their constant encouragement, and especially Hector for always supporting me, even through the most challenging moments.