THE UNIVERSITY OF CHICAGO

DEMOCRATIZING ACCESS TO EXTENSIVE CLIMATE DATASETS VIA DEEP LEARNING-POWERED TECHNIQUES

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE PHYSICAL SCIENCES
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE

BY
TAKUYA KURIHANA

CHICAGO, ILLINOIS
MARCH 2024

For Ko Kurihana (2/14/1939 – 6/23/2021), my beloved grandmother

”The best way to predict the future is to invent it.” – Alan Kay

Table of Contents

List of Figures

List of Tables

ACKNOWLEDGMENTS

I express my immense gratitude to my advisor Dr. Ian Foster for his continuous support with great patience of PhD study. His support and guidance for this interdisciplinary project always inspired me and helped me to pursue my research. Without the precious support it would not be possible to conduct this thesis.

I sincerely thank my thesis committees Dr. Elisabeth Moyer and Dr. Rick Stevens for their insightful feedback and guidance.

Finally, I would like to thank all members of Globus Labs for the support and encouragement.

ABSTRACT

Artificial Intelligence (AI) for science (AI4Science) develops cutting-edge AI algorithms and High-Performance Computing (HPC) to advance the frontier of science through the discovery of new scientific knowledge. However, hundreds of terabytes to petabytes of vast volumes of climate science datasets, including multispectral satellite instruments and numerical simulations, pose a challenge to processing and extracting useful information: Satellite instruments have captured cloud structure, size distributions, and radiative properties at a near-daily cadence over the past decades. High-resolution numerical climate and weather simulations have contributed to understanding the complicated interactions and feedback of Earth systems. These observations and simulations should help understand cloud and climate responses, but the complexity and size of this dataset have left it under-utilized. To aid the challenge and achieve the democratization of large volumes of climate science datasets by lowering a barrier to access to the core data, neural network-based approaches using self-supervised and supervised learning are promising solutions.

In this thesis, I first introduce rotationally invariant cloud clustering (RICC) that combines rotationally invariant autoencoder and hierarchical agglomerative clustering to generate unique new AI-generated cloud classes. Clusters produced from RICC detect meaningful distinctions between cloud textures, using only raw multispectral imagery as an input without reliance on location, time, derived physical properties, or pre-designated class definitions. Having RICC, I create a unique new cloud dataset, the AI-driven cloud classification atlas (AICCA), which clusters 23 years of ocean images from the Moderate Resolution Imaging Spectroradiometer (MODIS) on NASA's Aqua and Terra instruments –190 million of roughly 100 km x 100 km patches (128 x 128 pixels) - into 42 AI-generated cloud class labels. AICCA translates 872 TB of satellite images into 56 GB of class labels, metadata, and 13 cloud physical parameters. I finally demonstrate that RICC and AICCA apply to build a universal workflow to evaluate the bias exhibited by simulated clouds from high-resolution cloud models.

I present various neural network-based methodologies designed to deliver compressed or upscaled resolution of climate datasets, catering to the varied requirements of climate analysis. Clustering autoencoder that incorporates an online clustering algorithm offers a data-driven climate classification approach without subjective definitions. This method addresses a computational limitation in off-line clustering in RICC and compresses 70 years of GFDL-ESM2G climate simulation at 0.125 spatial resolution over the Continental United States under multiple warming scenarios, reducing it to a lower-dimensional space by a factor of 660,000. Additionally, I develop a physics-informed generative adversarial network utilizing self-attention computation to capture three-dimensional weather dynamics. The network super-resolves the three-dimensional wind data by upscaling the resolution by a factor of nine, ultimately aiming to provide accurate tracing and monitoring of greenhouse gas emissions. The super-resolution generative model compensates for the absence of high-resolution simulation outputs and saves computational time.

CHAPTER 1

INTRODUCTION

In an era marked by advancements in climate models and satellite instruments, a growing volume of data is being generated at the Petabytes (PB) scale, offering finer spatial and temporal resolutions [152, 177]. This proliferation of large scientific datasets enables scientists to harness the forefront of high-resolution data for various scientific applications. However, it is difficult to extract insight from complex patterns in such PB-scale datasets without access to a large computation resource, limiting the opportunities to leverage their rich information for users.

Over the past several decades, advancements in satellite-borne remote sensing instruments have produced petabytes of global multispectral imagery that capture cloud structure, size distributions, and radiative properties at minutes to daily cadence [173]. Meteorologists have developed a variety of cloud classification schemes [131, 180], which divide the various forms of clouds into four to several dozen of deterministic or nondeterministic types of cloud classes based on the texture, height, and thickness of clouds, as well as their surrounding atmospheric environment. Cloud classifications help the understanding of these cloud behavior, which plays a substantial role in the Earth's radiation budget by both reflecting sunlight and trapping infrared radiation. However, these enormous datasets are underutilized because climate scientists cannot in practice manually examine them to analyze spatial-temporal patterns. Here, automated cloud classification methods, which automate the classification of clouds by leveraging recent advancements in computing capability and AI technologies, can address the challenge.

Existing classification schemes are necessarily simplistic. The most standard classification, the ISCCP (International Satellite Cloud Climatology Project) schema, simply defines a grid of nine global classes based on low, medium, or high values of cloud altitude (cloud top pressure) and optical thickness [131, 132, 133]. Because this classification is typically applied pixel by pixel, it cannot capture spatial structures and can produce an incoherent spatial

distribution of cloud types in cloud imagery. The World Meteorological Organization's International Cloud Atlas [180], a more complex but subjective cloud classification framework, defines 10 basic classes and at most 100 of sub-classes with a complex coding procedure. The schema is subjective and difficult to automate and furthermore does not capture the full diversity of important cloud types. For example, it does not distinguish between open- and closed-cell stratocumulus clouds, placing them both in "stratocumulus," though the two have different circulation patterns, rain rates, and radiative effects [179]. Because the human eye serves as a sensitive tool for pattern classification, human observers can in principle group clouds into a larger set of types based on texture and shape as well as altitude and thickness. In practice, however, it has been difficult to devise a set of artificial cloud categories that encompass all cloud observations and can be applied consistently by human labelers. Moreover, the diversity of cloud morphologies and textures, and their multi-scale properties, makes classifying them into meaningful groupings a difficult task.

These issues motivate the application of AI-based algorithms for cloud classification. Most work to date on automated cloud classification has involved *supervised learning*, whereby ML models are trained to classify cloud images based on a training set to which humans have assigned labels. Early work on machine-based cloud classification algorithms integrated simple statistics with machine learning algorithms [11, 81, 175, 176], and combined textual and cloud physical features [136, 137]. More recent work [90, 126, 145, 183, 187, 188, 189, 192] takes advantage of convolutional neural networks (CNNs) [74, 146] reinforced by increasingly powerful modern computing hardware to achieve high classification accuracy in extracting relative features from images in cloud classification. However, supervised methods cannot discover unknown cloud types that may be relevant to climate change research because of their drawbacks: First, it requires a large annotated dataset. Conventional supervised cloud classification studies employed human labelers from a couple of participants to a few dozen of experts (e.g., 67 participants [126]) to clip a sufficient amount of training images. Second, because human-defined cloud classes are only well-defined for classic examples, which

account for just a small fraction of large satellite datasets [46, 133], supervised approaches fall short when used to classify diverse real-world data. Third, as labels are restricted to prior assumptions, they cannot identify cloud types that were not specified in the training data but that might be relevant to climate research. The difficulty of generating meaningful and consistent labels is a constant problem, and supervised learning approaches are the most successful when used on limited datasets containing classic examples of well-known textures. For example, Rasp et al. [126] classified just four particular patterns of stratocumulus defined and manually labeled by Stevens et al. [153].

To address the intrinsic drawback of the supervised learning approach for cloud classification and then serve the needs of climate research free from assumptions that may limit novel discoveries, the more appropriate choice is *self-supervised learning*, in which unknown patterns in data are learned without requiring predefined labels. The first demonstrations of self-supervised neural network methods applied to cloud images were made in the 1990s [160, 167]. Even with the primitive neural networks then available, Tian et al. [160] showed that cloud images from the GOES-8 satellite could be sorted automatically into ten clusters that reproduced the ten 'basic' WMO classes with 65–75% accuracy. Advances in deep neural network (DNN) methods enable to capture more complex object features from images. Denby [31] prototyped self-supervised cloud classification algorithms that used convolutional neural networks (CNNs) and produced cloud classes from the resulting compact representations via hierarchical agglomerative clustering (HAC) [63]. The works used only 12 classes successfully produced reasonable groups for different structures and textures of low clouds from near-infrared images from the GOES satellite in the tropical Atlantic. Yet, self-supervised approaches are challenging because they reveal underexplored associations between known cloud categories and cloud clusters generated as well as the lack of ground truth against which to compare the outputs, making evaluation a challenging task.

Applications of self-supervised deep learning and clustering are not limited to cloud imagery. They can bring benefits to the rapid evaluation of climate impact assesment

for a large volume of climate simulation outputs with high spatial-temporal resolution by generating a compact form of the core data. Climate classification – or identifying similar climatic regions or zones – has been used to understand the spatial variability of climate across a large area or facilitate the assessment of the climate change impact. Widely used climate classifications are often relied on deterministic definitions by human experts subjective based on prescribed thresholds [71, 117, 13, 25]. Thus, similar to self-supervised cloud clustering, self-supervised learning-based climate classification fundamentally diminishes the dimensionality of extensive climate simulation data into a defined set of zones. We can then understand the climate spatiotemporal patterns at a particular region or location without querying large climate datasets.

Despite the delusion of a vast volume of climate data, there are not always available appropriate resolutions of dataset for specific use cases, which require information of a higher-resolution of climate and weather dynamics due to computational resources and time constrain. Upscaling the resolution of climate data (i.e., 'Downscaling' technique in climate science, or 'super-resolution' in computer vision) can fill the gap between global and larger spatial-scale as well as between regional and local weather phenomena. Statistical downscaling depends on the availability of observation and assumption where a statistical relationship between variables holds over time [134]. Super-resolution is a generic computer vision technique to solve the case [80]. Deep neural network-based downscaling can bypass these constrains and offer the reduction of computational powers for generating high-resolution dataset. [170, 151, 184, 78, 83, 190, 2] Ultimately, the neural network-based downscaling technique allows more climate scientists and users to access the detailed finer-resolution of dataset without the expert knowledge and expensive computational resources, leading to democratizing climate dataset.

1.1 Thesis Statement

> Democratizing access to extensive climate datasets via deep learning-powered techniques.

Advancements in improving resolutions and systems of satellite remote sensing instruments, High-Performance Computing (HPC) platforms from hardware design to their computing powers, and cloud-resolving high-resolution climate and weather simulations have generated hundreds of terabytes to petabytes of vast volumes of climate science datasets. These extensive amounts of climate datasets make the extraction of meaningful data impractical for users, or require a solid amount of computational resources. Such a data deluge hampers scientists to fully use the information for science applications instead of catering more insights from more detail of information, as well as limits the availability of data in the era where open science is advancing. Simultaneously, the emulation of physical mechanisms of high-resolution data is essential for scientists as they are not always available due to the computational resources. Given that neural networks excel in learning nonlinear features in input data, I posit that a data-driven system capable of extracting important information from the large volumes of climate science datasets can offer opportunities to novel science discoveries.

Thus, I develop neural network-based approaches for *democritization* - or extraction and/or conversion of scientifically useful information to meet the user's requirement without expensive HPC resources - particularly in climate science applications to make complex datasets available for the climate and computer science community in this thesis.

1.2 Thesis Contributions

The primary contribution of this thesis is to propose various neural network approaches designed to deliver compressed or upscaled resolution of climate datasets for further processing and extracting scientifically useful information. The first part of the thesis consists of a self-supervised cloud classification framework that creates a new unique AI-driven cloud label and physical properties dataset. The method enables to discover unknown cloud types relevant to climate change research, with definitions based on spatial distributions, to deliver these complex datasets in compact forms that allow interpretation by facilitating access to

core data, and finally to apply the framework and dataset to validate simulated clouds from climate model outputs. The second part of the thesis focuses on numerical climate model outputs: I describe a self-supervised climate classification approach that encompasses an online clustering algorithm into the loss function to achieve computationally efficient clustering as well as group homogeneous climate zones delineated by different meteorological conditions without subjective definitions. The thesis finally introduces a super-resolution neural network that utilizes a self-attention computation and a generative model. It enables combining 3D dynamics of weather systems that are essential to reconstructing physically representative 3D wind fields for high-fidelity outputs. My contributions in more detail are as follows.

1. RICC: Rotationally Invariant Cloud Clustering. I first develop an automated cloud classification system that groups cloud images with similar texture and physical features as well as is agnostic to the orientations via a combination of the rotational-invariant loss function with the autoencoder and hierarchical agglomerative clustering. Existing classification schemes are necessarily simplistic due to artificial definitions. The most standard classification, the ISCCP (International Satellite Cloud Climatology Project) schema, simply defines a grid of nine global classes based on low, medium, or high values of cloud altitude (cloud top pressure) and optical thickness [131, 133, 132]. Since ISCCP is applied pixel by pixel, it cannot capture spatial structures and can produce an incoherent spatial distribution of cloud types in cloud imagery. The World Meteorological Organization's International Cloud Atlas [180], a more complex cloud classification framework, defines 28 different classes (of which 10 are considered 'basic types') with a complex coding procedure that depends on subjective judgments. The schema is subjective and difficult to automate, and furthermore does not capture the full diversity of important cloud types nor scale to the variety of images for annotations by human experts. I find that RICC generates clusters that are meaningful aspects of cloud physics, appropriately spatially coherent, and invariant

to orientations of input images. The design of a series of evaluation protocols helps if the resulting cloud clusters can be scientifically useful in assigning operational classifications to cloud images.

2. AICCA:AI-driven Cloud Classificaiton Atlas. Next, I create a novel self-supervised learning-based climate dataset that composites 23 years of cloud labels and associated metadata based on the global multispectral images from the Moderate Resolution Imaging Spectroradiometer (MODIS) instruments on NASA's Aqua and Terra satellites [102]. Advancements in satellite-borne remote sensing instruments can provide few hundred of terabytes to petabytes of dataset with researchers to study cloud structure, size distributions, and radiative properties. While understanding trends in cloud behavior is arguably the principal challenge in climate science, these enormous datasets are underutilized because climate scientists cannot in practice manually examine them to analyze spatial-temporal patterns. Instead, some kind of automated algorithm is needed to identify physically relevant cloud types. AICCA addresses the scientific issue by combining AI-generated cloud labels from RICC and processing MODIS cloud products to composite associated metadata. Finally, AICCA translates 872 TB of satellite images into 26.7 GB (patch-level Section 4.4) and 54 GB (daily-level Section 4.4) of class labels and cloud properties.

3. Self-supervised Cloud Biass Assessment. Increasing computing power means the spatial scale of climate simulations has shrunk to the point where their output can resolve complex cloud textures and physical mechanisms [110]. A combination of self-supervised deep learning and clustering could help in assessing whether models capture those textures correctly. The approach is applied to group diagnosis of the variety of convection patterns among major models [106], but hasn't yet been used to evaluate the model representation of clouds against satellite observation. Therefore, I build the Self-supervised Cloud Biases Assesment (SCuBA), a framework that applies RICC to synthesized radiances from numerical climate and weather models.

4. Clustering Autoencoder for Climate Classification. Climate model simulation outputs are increasingly used for practical applications and more granular decision making such as climate resilience assessments [92] at specific regions or for specific issues. This large volume of simulation outputs with high spatial-temporal resolution overwhelm the capacity of computing powers and resources when climate scientists and practitioners analyze trends and mechanism under warming climate scenarios. Furthermore, the rapid evaluation of climate impact assessment may downturn due to the increasing volume of climate simulations with a variety of ensemble of models and scenarios [118]. Therefore, I develop clustering autoencoder algorithm that dimensionally reduces the vast volume of spatiotemporal numerical climate projection data into a compact representation to group unique climate patterns in a data-driven fashion. The clustering autoencoder includes clustering step, which is in general performed in off-line of training of autoencoder, as a part of loss function by adapting an online clustering algorithm so that the resulting clusters can embed information from a larger source of dataset with less computation overhead. Resultant climate zones can help the evaluation of climate patterns immediately without querying large climate datasets.

5. Physics-informed 3D Super-Resolution GAN. Accurate racing and monitoring of greenhouse gas (GHG) sources is a key measurement to take action for tackling the mitigation of global warming issues [30]. The higher spatial resolution of wind simulation enables precise tracking of GHG emissions from potential sources, and helps decision-making in various fields such as policymakers, agriculture, and renewable energy. However, high-resolution simulation is not always available due to the demands of computational resources. Super-resolution (SR) is one of the solutions to achieve the goal: conventional SR techniques rely on convolutional neural networks (CNNs) to reproduce realistic high-resolution wind fields [151, 184, 78, 83, 190, 2]. However, these CNN-based approaches either perform on 2D data or fall short of capturing the 3D dynamics in weather systems because a convolutional kernel truncates the association between vertical layers, limiting to learning of convection

and diurnal cycles induced by incoming solar radiations. To aid the challenge, during an internship at IBM Research at Yorktown, I develop a physics-informed super-resolution SR generative adversarial network that super-resolves the three-dimensional 3D low-resolution wind fields by a factor of nine to address the absence of high-resolution climate data due to the demands of computational resources.

1.3 Thesis Organization

The thesis is organized as follows. Chapter 2 provides fundamental related works in cloud classification, autoencoder, and transform-invariant for RICC and AICCA. Chapter 3 describes the algorithm of rotationally invariant cloud clustering that clusters multidecadal MODIS cloud imagery to extract scientifically meaningful groups of clusters. I present a series of evaluation protocols to define the usefulness of clusters in cloud study by constructing rigorous examinations of clusters. Chapter 4 presents the methodology to assign clusters from rotationally invariant cloud clustering and defines the structure of AI-generated cloud datasets. I investigate the exploratory power of the dataset to represent richer information than the existing artificial classifications do, and show the use cases to gain insights in cloud responses. In the second part of the thesis, I elucidate research outcomes that contribute to the generation of scientifically valuable climate datasets through the application of AI-powered techniques as follows. Chapter 6 describes the algorithm of clustering autoencoder that extracts important input information into a low-dimensional embedding as well as trains an online clustering algorithm as a part of loss function to group similar features of data into a predefined number of clusters in which I evaluate the physical reasonableness and similarities against existing climate zones. Chapter 7 presents a novel neural network architecture to embed three-dimensional dynamics of weather systems for a super-resolution generative adversarial network.

CHAPTER 2

CLOUD CLASSIFICATION

This chapter is dedicated to reviewing literature oni mportant components that require to build a rotationally invariant cloud clustering algorithm.

2.1 Standard Cloud Classification

A cloud classification algorithm classify the various forms of clouds into deterministic or nondeterministic groups since 19 century when Luke Howard first proposed three categories of clouds – stratus, cumulus, and cirrus. The World Meteorological Organization (WMO) established the cloud classification [180] shown in Table 2.1. This classification is widely acknowledged across scientific and non-scientific communities, becoming a de facto standard for cloud types. Most cloud classification work relies on the WMO's definition to develop algorithms for training, testing, and evaluation. The classification defines ten main groups, called *genera*: Cirrus (Ci), Cirrocumulus (Cc), Cirrostratus (Cs), Altocumulus (Ac), Altostratus (As), Nimbostratus (Ns), Stratocumulus (Sc), Stratus (St), Cumulus (Cu), and Cumulonimbus (Cb), each characterized by *height* and *texture* information, mainly from ground-based observations. The ten genera are further divided into sub-groups such as *species* and *varieties* to classify intermediate types of clouds.

Another approach to cloud classification is to use data from satellite passive sensors and ground-based remote sensors, including lidar, radar, and radiometers. For example, the International Satellite Cloud Climatology Project (ISCCP) cloud-type classification [131, 137] shown in Figure 2.1 identifies nine cloud types: Cumulus, Stratocumulus, Stratus, Altocumulus, Altostratus, Nimbostratus, Cirrus (and Cirrostratus), and deep convective cloud. Their cloud types represent the meteorological relationship between two cloud physical parameters, cloud top pressure and cloud optical thickness, which they use to deterministically classify cloud types at the pixel level.

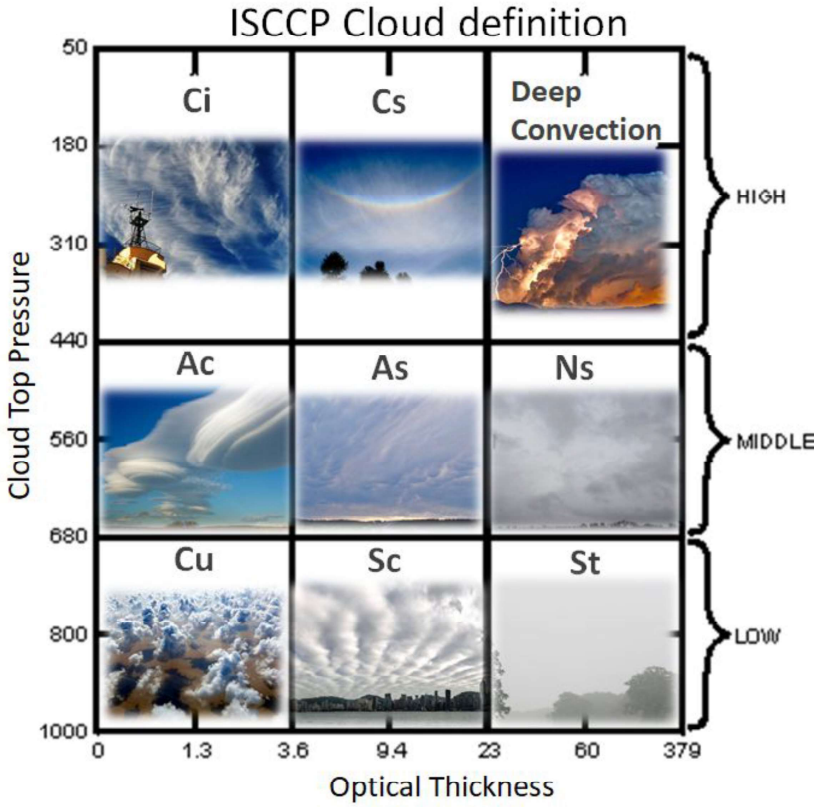


Figure 2.1: ISCCP cloud classification illustrating the categorization of clouds based on their optical properties, spatial distribution, and vertical structure based on high/middle/low categories respectively. The classification system provides a comprehensive framework for understanding and analyzing different cloud types, aiding in meteorological and climate studies

Table 2.1: Cloud classification by WMO [180].

Level	Name	Description	Height		
			Polar	Mid-lat	Tropical
High	Cirrus (Ci)	Detached clouds; fibrous or silky sheen	3–8 km	5–13 km	6–18 km
	Cirrocumulus (Cc)	Thin, white patch sheet or layer cloud composed of very small elements			
	Cirrostratus (Cs)	Transparent, whitish cloud veil of fibrous or smooth appearance			
Middle	Alto-cumulus (Ac)	White or grey path, sheet or layered cloud with shading composed of laminae	2–4 km	2–7 km	2–8 km
	Altostratus (As)	Greyish cloud sheet or layer of striated, fibrous or uniform appearance			
	Nimbostratus (Ns)	Dark grey cloud layer with rain or snow, blot out the Sun			
Low	Stratus (St)	Grey cloud layer with fairly uniform base	Surface to 2 km	Surface to 2 km	Surface to 2 km
	Stratocumulus (Sc)	Grey or whitish clouds, composed of non-fibrous tessellations or rounded masses			
	Cumulus (Cu)	Detached clouds, generally dense and with sharp outline, developing vertically			
	Cumulonimbus (Cb)	Heavy and dense cloud, with considerable vertical extent, in the form of a mountain or huge towers			

2.2 Automated Cloud Classification

Automated cloud classification has been studied since the 1970s, developing from simple statistics with machine learning algorithms and shallow neural networks in the 1990s, to neural network-based technologies after the mid-2010s. The motivation for building an automatic system originates in the 1970s [42] with the launch of multi-band geostationary and other meteorological satellites that allowed climate researchers to leverage richer observation data for further understanding of cloud feedback. The resulting datasets, however, are too large to be categorized cost effectively by human experts: instead, they must be processed by machine-power.

Most work to date on automated cloud classification has involved supervised learning, whereby a classifier is trained with many images of clouds labeled with the classes that one seeks to recognize classes in ISCCP, WMO, and/or any artificially defined classes. Early work on machine-based cloud classification algorithms integrated simple statistics with machine learning algorithms [11, 81, 176, 175], and combined textual and cloud physical

features [136, 137]; more recent work [126, 183, 187, 192] takes advantage of state-of-the-art deep learning technologies reinforced by increasingly powerful modern computing hardware. Such approaches have the significant advantage of being easy to evaluate, assuming that a reliable and sufficiently large collection of ground truth data is available.

Wood and Hartmann [178] and Muhlbauer et al. [107] target mesoscale organization (1–100 km scale), which plays an important role in the Earth's energy budget, to extend the scope of classification work from understanding the distribution of major cloud types to investigating the relationship between the complex textures of low clouds and the physical characteristics found within common mesoscale cloud organizations. Wood and Hartmann [178] annotate four frequent mesoscale cloud patterns associated with the existence of open- and closed-cell structures, where a cell is a tightly connected hexagonal cloud structure that if "open" has an empty center that is empty and if "closed" has a dense center—No Mesoscale Cellular Convection (MCC), Closed MCC, Open MCC, and Cellular but disorganized—that are observed by the MODIS Terra satellite. Recent similar work by Rasp et al. [126] focuses on shallow marine clouds to train deep learning models and classifies them into new four categories, where Stevens et al. [154] identify the four recurrent categories of mesoscale organization from MODIS Terra and Aqua satellites: Sugar: dusting of very fine clouds, with little evidence of self-organization; Flower: large-scale stratiform cloud features appearing in bouquets, well separated from each other; Fish: large-scale skeletal networks of clouds separated from other cloud forms; and Gravel: meso-beta lines or arcs defining randomly interacting cells with intermediate granularity from their definition. The classification analysis reveals that their customized annotation, whereby human labelers classify clouds according to differences in cloud texture, exhibits physically meaningful cloud regimes when they compare temperature, relative humidity, and vertical velocity across the four classes.

2.3 Limitations of Supervised Learning Approaches

Supervised cloud classification has drawbacks. First, it requires a large annotated dataset. Second, because human-defined cloud classes are only well defined for classic examples, which account for just a small fraction of large satellite datasets [133, 46], supervised approaches fall short when used to classify diverse real-world data. Third, as labels are restricted to prior assumptions, they cannot identify cloud types that were not specified in the training data but that might be relevant to climate research.

These difficulties have been highlighted by several studies. When Wood and Hartmann [178] investigated the relationship between the complex textures of low clouds and the physical characteristics found within common mesoscale (1–100 km) cloud organizations, they found that four frequent mesoscale cloud patterns associated with the existence of open- and closed-cell structures, often classified as stratus or stratocumulus in the ISCCP classification, occur in different geographical regions and have different distributions of liquid water path, suggesting that the spatial variability of low clouds is underrepresented in the standard cloud types.

2.4 Unsupervised Neural Network for Cloud Classification

Unsupervised learning, one of the AI algorithms, learns patterns in input data without either predefined labels or artificial interventions to group similar patterns of data. Few attempts have been made to explore the potential of unsupervised learning for cloud classification.

An early and widely used technique involves the use of a self-organizing map (SOM) [69]. SOM technique produces dimensionality-reduced representations while preserving topological relationships in input data via a two-layer (i.e., input and output layers) neural network. The network first selects the two largest eigenvalues to span a two-dimensional space that is composed of *nodes* with initial weights. Each data point in the dataset has its Best Matching

Unit (BMU) calculated based on the shortest Euclidean distance and is then considered matched to that node. The node and nodes that fall within a defined neighborhood are updated to better match the assigned point until a given step. Early studies [160, 167] uses SOM to classify satellite imagery in both visible and infrared channels into basic cloud types (e.g., stratocumulus, cumulus) and land and/or ocean background types without clouds. Downstream classification task performs comparisons of the accuracy with supervised neural networks, revealing that unsupervised learning may not match the artificial cloud categories defined by climate scientists. The result at the early stage indicates that evaluation metrics used for supervised learning may not be suitable for assessing the heuristic power of unsupervised learning. Recent studies using SOM, motivated by improving unsupervised cloud classification performance from simply applying clustering algorithms to two-dimensional cloud optical thickness (COT) – cloud top pressure (CTP) joint histogram [58], show that clusters produced using the SOM technique from ISCCP data [97, 98] and MODIS prodacts [140, 141] have been used to investigate various properties associated with cloud clusters such as radiation, vertical velocities and cloud phase.

The use of deep neural networks for unsupervised cloud classification opens up a new research direction: Denby [31] applied representation learning techniques to classify mesoscale cloud organizations. In representation learning, CNNs are trained to efficiently embed meaningful features in the input data on a latent representation, which is then used for a clustering or regression protocol. Denby trained a 34-layer residual neural network (ResNet-34) on GOES-16 images to formulate the representation, and then applied hierarchical clustering to the latent representation of unseen GOES-16 images. The resulting clusters differentiate cloud images based on the strength of radiation in visible channels (band 1 of GOES-16); the triplet loss applied to the ResNet-34 network encourages the separation of dissimilar texture of images. However, their example classes still contain similar cloud images across several clusters, e.g. closed-cell stratocumulus clouds are distributed among at least four classes in their Figure 2 [31], indicating a limitation in their ability to classify

small patches of large cloud structures by unsupervised learning.

2.5 Autoencoders

An autoencoder (AE) [51, 52, 72], a widely accepted unsupervised learning method, combines an encoder, decoder, and loss function to first encode input data into a compact lower-dimensional *latent representation* and then to decode that representation at the bottleneck to outputs, in a manner that minimizes loss function that quantifies the difference between input and output. Encoder E extracts essential information from high-dimensional inputs into lower-dimensional intermediate layers. The latent representation learns the efficient data representation of the inputs via dimensionality reduction. Instead, decoder D restores the original data from z, a latent representation at the bottleneck layer, where the input data $X = \{x_1, \cdots, x_n\}$ are encoded as $z = E(X)$, such that $\hat{X} = D(z)$. The *reconstruction loss* function measures the reconstruction error while autoencoder approximates $\hat{X}$ in high fidelity in the training process, but it typically cannot perfectly restore the inputs X. During the optimization process, the loss function $\mathcal{L}$ used in common autoencoders minimizes the squared difference between inputs and outputs as a measure of the restoration accuracy as follows:

$$\mathcal{L} = \sum_{x \in X} ||x - \hat{x}||_p^p, \tag{2.1}$$

where $||\cdot||_p$ denotes the p-norm of the inputs and the restorations. Autoencoder typically uses a symmetric encoder-decoder architecture, while a recent study [48] shows asymmetric encoder-decode, which deploys a relatively larger encoder and a lightweight decoder small enough to reproduce input images, performs compatible classification accuracy by a large supervised vision transformer.

As autoencoders can preserve only essential information in the latent representation, by leaving out noise in the inputs, they are often used for an anomaly detection [135], denoising [165], and image inpainting [115, 155]. In addition, autoencoders have been

applied successfully to a wide variety of image recognition problems [18] via clustering and classification of the compact representation.

In current AI4science literature, autoencoder generally implies convolutional autoencoder (CAE) [91], which has been widely applied in a wide range of scientific papers because recent advancements in computer vision have exhibited impressive classification performance via convolutional neural networks in image recognition [74, 146]. A CAE substitutes a fully connected layer used by an orthodox fully connected encoder-decoder network with a convolutional layer so as to preserve spatial sub-patterns in input images. Racah et al. [122] firstly introduce the use of CAE in climate science applications where they trained a CAE on climate simulation output data to detect extreme weather invents. Their network optimized a semi-supervised loss function to predict a bounding box where events are likely to trigger, generating a representation that can be used to better understand complex large-scale climate datasets. Despite that CNNs are widely adapted to autoencoder, the use of a vision transformer at an encoder part of autoencoder emerges as a new architecture [6]. Overall, an autoencoder is designed such that the training of the entire network generates a model capable of restoring the principal patterns of the original structure in an unsupervised manner.

Variational autoencoder (VAE) [68] is another popular autoencoder variant [1]. In contrast to the unconstrained optimization of a regular autoencoder, a VAE learns how to transform input data in a probabilistic manner into a low-dimensional latent representation from which it can generate the approximation of any image. VAE approximates a continuous latent space from input data that is assumed to be isotropic gaussian distributions via their mean and variance.

1. VAE is also seen as **generative model**, which generates synthetic data from the probability distribution function, generally assumed to be a Gaussian distribution, at the latent representation via the learning process. In this thesis, we do not consider VAE as an alternative approach because our task is simply to reduce the dimension of satellite images to a latent representation.

2.6 Transform Invariant

Transform invariant representation learns a robust mapping in terms of the translation, scale, and/or orientation of features from input images throughout training neural networks. That is, a transform invariant network can map a relevant feature from high dimensional input data to the same low dimensional latent representation regardless of its shift, scale, or orientation. This learning of transform-invariant features may be achieved in three different ways: with a customized loss function, a sampling and data augmentation approach, and a specialized transform architecture/module.

The most explicit approach to achieving transform invariant feature learning is customizing a loss function to generate invariant representations. Matsuo et al. [95], Matsuo and Shimada [94] propose a shift-invariant autoencoder to independently learn typical sub-patterns of a given input and transform those sub-patterns into a consistent invariant representation. The loss function minimizes the sum of differences between restoration images with and without the transform operator, so that the optimization of the encoder-decoder pair finds a consistent latent representation for all orientations of each input image. Cheng et al. [23] introduced a loss function that combines a rotation-invariant and a Fisher discrimination regularizer to explicitly impose the rotation-invariant feature on the latent representation. Rotaion-invariant regulariser averages over differences between an original image and the multiple rotated version of the input to approximate invariant to any rotation of images. Fisher regulariser plays as like contrastive learning term such that latent representations from the same class minimizes the intra-class variations but maximize inter-class distance. Lohit and Trivedi [85] calculated the maximum of spherical cross-correlation for spherical 3D images in the loss function.

Shen et al. [144] applied a combination of random scale, rotate angle, and translation proportion to each local feature map, a receptive field collected from a subset of an input image at a given kernel resolution, and for this operation, a convolutional neural network achieves transform invariant features. Benton et al. [14] proposed a general transform

invariant framework that parameterizes a distribution over the set of affine transformations of data including translations, rotations, and scales to learn invariance features from a variety of data augmentations. Chaman and Dokmanic [20] introduced adaptive polyphase sampling (APS), a simple sub-sampling scheme that allows convolutional neural networks to achieve 100% consistency in classification performance under shifts, without any loss in accuracy. In satellite imagery classification context, Jain et al. [57] discussed that Bootstrap Your Own Latent (BYOL) approach [43], a type of self-supervised neural network that learns two different views produced by different augmentations of an identical image, encourages learning the semantic aspect of input data, leading to obtaining invariant feature in the latent representation. A similar and straightforward idea is introduced by Dieleman et al. [33] by concatenating representations through a combination of multiple cropped viewpoints with random rotation of galaxy images to improve the performance of downstream classification casks regardless of the orientation of input galaxy images.

Many previous attempts have introduced transform invariant architectures into a neural network in order to obtain a specific invariant feature explicitly. Jansson and Lindeberg [59] designed the Foveated scale-channel network, which has a fixed size receptive filed to simultaneously process an original image and its rescaled images in different channels to acquire a scale-invariant feature for identical object in images. Sohn and Lee [149] achieved a translation and scaling invariant feature representation through probabilistic max pooling, which selects a transformed projection through the filter across the set of transformations. Anti-aliased max pooling [195] subsamples feature maps from a naive max-pooling by a stride size, which achieves shit-invariance via a max pooling operation. Farabet et al. [36] transformed a raw input image through a Laplacian pyramid to obtain shift-invariant features. Kanazawa et al. [65] proposed a locally scale-invariant convolutional layer where a pyramid of different scaled images are fed to convolutional operation and finally take max-pool over different scales to achieve transform invariance. Baccouche et al. [8] proposed a sparse shift-invariant autoencoder that introduces a translation vector to

build code into the shifted version of input images. The training process uses an additional variable to find the best scale shift translation. Jaderberg et al. [56] introduced a spatial transformer, a learnable module applied in the network to obtain spatial transform feature maps. The transformer module learns the invariant features through a resampling from the transformed grid, which takes localized features from the input images.

CHAPTER 3

RICC: ROTATIONALLY INVARIANT CLOUD CLUSTERING

The rotationally invariant cloud clustering algorithm uses multispectral satellite imagery from NASA's Moderate Resolution Imaging Spectroradiometer (MODIS) [64] collected by Aqua and Terra instruments in developing and evaluating our rotationally invariant autoencoder and hierarchical agglomerative clustering (RICC)-based unsupervised learning system for cloud classification.

3.1 Data Collection

NASA's MODIS Aqua and Terra satellites have observed 36 spectral bands of range 0.4–14.4 μm (i.e., visible to thermal infrared) radiances since 2002 (Aqua) [103] and 2000 (Terra) [101] through to the present. These instruments collect data over an approximately 2330 km by 2030 km *swath* spatial coverage every five minutes, generating the MODIS Level 1B calibrated radiance product (MYD021KM/MOD021KM: hereafter, **MOD02**)[1] in Hierarchical Data Format (HDF) files that each contain 2030 pixels $\times$ 1354 pixels $\times$ 36 bands. Spatial resolution for pixels directly beneath the observational instruments is 1 km, degrading to at most several km for other pixels at an angle of the instrument. However, the frequent scanning by instruments overlaps consecutive observations, allowing us to approximate the spatial resolution of pixels to 1 km and at most 2 km.

There are six spectral bands that are most relevant for cloud top and optical properties. Bands 6, 7, and 20 relate to cloud optical properties (e.g., cloud optical thickness and effective radius), and bands 28, 29, and 31 relate to the separation of high and low clouds and liquid and ice particle phases (e.g., cloud top pressure and cloud phase). However, for the Aqua instrument, we use band 5 as an alternative to band 6 due to a known stripe noise issue in

1. NASA uses the prefixes MYD and MOD to distinguish between data observed from Aqua and Terra, respectively

Aqua band 6 [124]. The use of bands 5, 6, and 7 are only limited to local daylight times, which restricts the amount of available swath to only about half of all archived MOD02 data. For the efficient learning of these cloud features on neural networks, the system chooses to use a smaller geographical unit, a *patch*, as the image for cloud classification. That is, this small subset of a typical MODIS image is a 128 pixels $\times$ 128 pixels $\times$ 6 band region subsampled from a 2030 pixel $\times$ 1354 pixel $\times$ 36 band MODIS image. The total number of swath images per band is (12 swath/hour) $\times$ (12 hour/day) $\times$ (365 day/year) $\times$ (21 + 23 years, for Aqua and Terra, respectively) $\approx$ 2.3 million swathes. The hypothesis in selecting specific wavelengths and spatial scales here is that the use of six bands is expected to obtain a useful association of cloud physical parameters used for ISCCP cloud classification to the resulting group of clusters, and a 100 km spatial scale is able to represent a group of cloud structures.

Aside from radiance data via MOD02, MODIS provides a variety of derived products from cloud physical parameters to vegetation index. MODIS06 level 2 cloud product at 1 km resolution (MYD06L2 and MOD06L2; hereafter, **MOD06**) [12, 119] provides cloud optical properties and cloud top properties. RICC employs the MOD06 variables only as a diagnostic, to evaluate associations between resulting cloud clusters and cloud physical properties; in particular, they are not included in our RICC training data, which are thus free from any assumptions made by ISCCP cloud classification. Additionally, the MODIS03 product offers pixel-level latitude and longitude data from the MYD03/MOD03 (hereafter **MOD03**) geolocation fields for each swath, allowing us to identify latitude and longitude location on swathes. All MODIS products are accessible via the NASA Level-1 and Atmosphere Archive and Distribution System (LAADS), grouped into per-swath files. I summarize the specific products used for the proposed classification framework in Table 3.1.

The data collection scheme of RICC for complete 23-year (2000–2022) patches from Aqua and Terra MODIS images requires the constraints that they 1) are disjoint in space and/or time; 2) include only ocean pixels, 3) each includes at least 30% cloud pixels, and 4) are

Table 3.1: MODIS products used to create the AICCA dataset. As noted in the text, each product name *MOD0X* in the first column refers to both the Aqua (MYD0X) and Terra (MOD0X) products. Source: NASA Earthdata.

Product	Description	Band	Primary Use	Section
MOD02	Shortwave infrared (1.230–1.250 μm)	5	Land/cloud/aerosol properties	§3.2
	Shortwave infrared (1.628–1.652 μm)	6	Land/cloud/aerosol properties	
	Shortwave infrared (2.105–2.155 μm)	7	Land/cloud/aerosol properties	
	Longwave thermal infrared (3.660–3.840 μm)	20	Surface/cloud temperature	
	Longwave thermal infrared (7.175–7.475 μm)	28	Cirrus clouds water vapor	
	Longwave thermal infrared (8.400–8.700 μm)	29	Cloud properties	
	Longwave thermal infrared (10.780–11.280 μm)	31	Surface/cloud temperature	
MOD03	Geolocation fields		Latitude and Longitude	§4.4
MOD06	Cloud mask		Cloud pixel detection	§3.2
	Land / Water		Background detection	
	Cloud optical thickness		Thickness of cloud	§3.5
	Cloud top pressure		Pressure at cloud top	
	Cloud phase infrared		Cloud particle phase	
	Cloud effective radius		Radius of cloud droplet	

applied to circular masking to stabilize optimization of neural networks. The resulting set comprises about 153 590 874 individual 128×128 pixel (∼100 km by 100 km) ocean-cloud patches, giving `OC-Patches`. From this set, I extract the following subsets for training and testing.

- $\texttt{OC-Patches}_{\texttt{AE}}$: 1 million `OC-Patches` used for training autoencoders. This is about 0.65% of the full 23-year (2000–2022).

- $\texttt{OC-Patches}_{\texttt{HAC}}$: A set of 74 911 `OC-Patches` used for identifying a set of k cluster centroids, $\mu = \{\mu_1, \cdots, \mu_k\}$ from the year 2003 (the first year in which both Terra and Aqua satellites ran for the entire year concurrently).

- $\texttt{OC-Patches}_{\texttt{HAC-2}}$ and $\texttt{OC-Patches}_{\texttt{HAC-3}}$: Additional version of 77 235 and 76 143 `OC-Patches`, to account for potential sources of bias from sampling: Because the specific days used in $\texttt{OC-Patches}_{\texttt{HAC}}$ may affect our results, we assemble two additional versions of $\texttt{OC-Patches}_{\texttt{HAC}}$, selecting two days without replacement from each season in 2003.

- `Test`: A set of 2000 `OC-Patches` used for evaluating whether resulting latent -

representations from a trained autoencoder achieve rotation-invariance features (see Section 3.5.4).

3.2 Rotation Invariant Autoencoder

An autoencoder [51, 52] comprises an encoder, used to map input images into a compact lower-dimensional latent representation, followed by a decoder, used to map that representation to output images. The loss function minimizes the difference between input and output. The resulting latent representation in the trained autoencoder both 1) retains only relevant features for the target application in input images, and 2) maps images that are similar (from the perspective of the target application) to nearby locations in latent space.

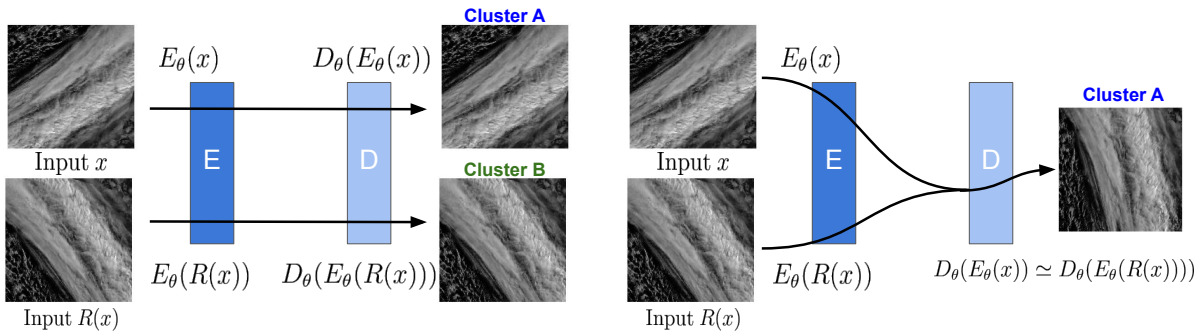
The loss function minimizes the difference between an original and a restored image based on a distance metric during autoencoder training. The most commonly used metric is a simple ℓ^2 distance between the autoencoder's input and output:

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{x \in S} ||x - D_\theta(E_\theta(x))||_2^2, \tag{3.1}$$

where S is a set of training inputs; $\boldsymbol{\theta}$ is the encoder and decoder parameters, for which values are to be set via training; and x and $D_\theta(E_\theta(x))$ are an input in S and its output (i.e., the restored version of x), respectively.

However, autoencoder optimizing with Equation 3.1 may generate different representations for an image x and the rotated image $R(x)$, as shown in Figure 3.1, with the result that the two images end up in different clusters. Cloud types can occur in different orientations as cloud formation is driven not by wind direction but by mechanisms such as adiabatic or non-adiabatic cooling, convection, advection, and terrestrial effects. Since any particular physically driven cloud pattern can occur in different orientations, it is inadequate for the autoencoder for a meaningful cloud classification purpose to produce different latent representations that depend solely on orientation and a clustering algorithm assigns different

clusters.



(a) Conventional autoencoder (b) Rotationally invariant autoencoder

Figure 3.1: Illustration of the learning process when training (a) a conventional autoencoder with Equation 3.1 vs. (b) a rotationally invariant autoencoder with Equation 3.2. Because a conventional autoencoder reflects orientation in the latent representation, two input images that are identical in texture but different in orientation are assigned to different clusters, A and B. The rotationally invariant autoencoder produces a latent representation that is agnostic to orientation, allowing clustering to group both together.

To address this rotation-dependence problem in autoencoder, we propose a rotation-invariant (RI) loss function that generates similar latent representations, agnostic to orientation, for similar morphological clouds (Figure 3.1b). This RI autoencoder, motivated by the shifted transform invariant autoencoder of Matsuo et al. [95], uses a loss function $\mathcal{L}$ that combines both a rotation-invariant loss, $\mathcal{L}_{\text{inv}}$, to learn the rotation invariance needed to map different orientations of identical input images into a uniform orientation, and a restoration loss, $\mathcal{L}_{\text{res}}$, to learn the spatial structure needed to restore structural patterns in inputs with high fidelity. The two loss terms are combined as follows, with values for the scalar weights λ_{inv} and λ_{res} chosen as described below:

$$\mathcal{L} = \lambda_{\text{inv}}\mathcal{L}_{\text{inv}} + \lambda_{\text{res}}\mathcal{L}_{\text{res}}, \tag{3.2}$$

where we design our training protocol in Section 3.3 to sweep over possible combinations of hyperparameter space to find the optimal combination.

The rotation-invariant loss function $\mathcal{L}_{\text{inv}}$ computes, for each image in a minibatch, the

difference between the restored original and a set of images obtained by applying a set $\mathcal{R}$ of multiple scalar rotation operators to the original image. We empirically observe that having a smaller angle interval helps convergence in optimization but a larger angle interval causes divergence in optimization. In our configuration, $\mathcal{R}$ rotates an input by every five degrees in the set {0, 5, ..., 355}:

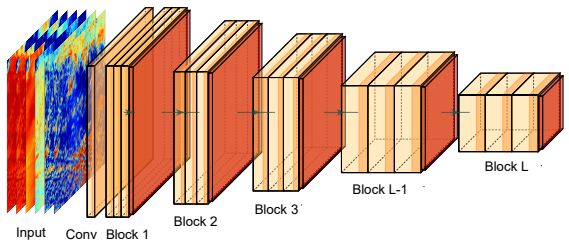
$$L_{\text{inv}}(\boldsymbol{\theta}) = \frac{1}{N} \sum_{x \in S} \sum_{R \in \mathcal{R}} ||D_\theta(E_\theta(x)) - D_\theta(E_\theta(R(x)))||_2^2, \tag{3.3}$$

Thus, minimizing Equation 3.3 yields values for $\boldsymbol{\theta}$ that produce similar latent representations for an image, regardless of its orientation.

The restoration loss, $L_{\text{res}}(\boldsymbol{\theta})$, learns the spatial substructure in images by computing the sum of minimum differences over the minibatch:

$$L_{\text{res}}(\boldsymbol{\theta}) = \sum_{x \in S} \min_{R \in \mathcal{R}} ||R(x) - D_\theta(E_\theta(x))||_2^2. \tag{3.4}$$

Thus, minimizing Equation 3.4 results in values for $\boldsymbol{\theta}$ that preserve spatial structure in inputs.



Layer	Shape	Operation
Input	128 x128 x 6	Resize
Conv	32 x 32 x 6	Conv x 1
Block 1	16 x 16 x 32	Conv x 3
Block 2	16 x 16 x 64	Conv x 3
Block 3	8 x 8 x 128	Conv x 3
Block 4	4 x 4 x 256	Conv x 3
Block 5	2 x 2 x 512	Conv x 3

Figure 3.2: Architecture of encoder part of RI autoencoder. Left: diagram of encoder that consists of L blocks, each with three convolutional layers (orange) activated by leaky ReLU, and with batch normalization (red) applied at the final convolutional layer in each block before activation. Right: summary of shape and operations per layer. RI autoencoder is composed of total 10 996 230 trainable parameters, which is approximately half the size of ResNet-50.

The neural network architecture is the other critical factor needed to achieve rotation

invariance: Following the heuristic approach of deep convolutional neural networks, we designed an encoder and decoder that stack five blocks of convolutions shown in Figure 3.2, each with three convolutional layers activated by leaky ReLU [108], and with batch - normalization [54] applied at the final convolutional layer in each block before activation. Shallower networks (i.e. stack fewer blocks of convolutions) are not capable to learn rotation-invariant features because a local receptive field does not cover global features sufficiently. We train our RI autoencoder with `OC-Patches` for 100 epochs by using stochastic gradient descent [79] with a learning rate of 10^{-2} on 32 NVIDIA A100 GPUs in the Argonne National Laboratory ThetaGPU cluster. Our training protocol reveals that the selection of an adequate learning rate is another essential factor.

3.3 Training protocol

The performance of the RI autoencoder on a particular training set is sensitive to the values assigned to the λ parameters. We formulate a grid search process for finding an optimal combination of λ_{inv} and λ_{res} as follows:

1. Fix λ_{res} and learning rate lr from 10^{-4} to 10^{-2}, for stochastic gradient descent. Set $\lambda_{\text{inv}} = 0$.

2. Train the autoencoder to obtain a baseline trained model $\mathbf{A}$; measure its restoration loss L_{res}, on a holdout set X_{holdout}: $L_{\text{res}}(\mathbf{A}, X_{\text{holdout}})$.

3. Set $\lambda_{\text{inv}} = 0.1$.

4. Train the autoencoder with the new λ_{inv} to obtain a new trained model $\mathbf{B}$; measure its restoration loss on the same holdout set, giving $L_{\text{res}}(\mathbf{B}, X_{\text{holdout}})$.

5. If the restoration loss of the newly trained model $\mathbf{B}$ is less than 20% larger than that of the baseline $\mathbf{A}$, i.e., if $L_{\text{res}}(\mathbf{B}, X_{\text{holdout}}) \leq 1.2 L_{\text{res}}(\mathbf{A}, X_{\text{holdout}})$, then double λ_{inv}.

6. Compare the results produced by $\mathbf{B}$ for images $x_i \in X_{\text{holdout}}$, each replicated N times, and with each replicate rotated by a different θ. Observe whether, for each image, all $\mathbf{B}(\mathcal{R}(x_i, \theta))$ have the same canonical rotation. (In our implementation, we assume that $N = 12$ and $\theta \in \{0, 30, \cdots, 330\}$.) We verified $\mathbf{B}(\mathcal{R}(x_i, \theta))$ based on the cosine similarity and eye-ball observation.

7. If for all $x_i \in X_{\text{holdout}}$, all $\mathbf{B}(\mathcal{R}(x_i, \theta))$ are the same, terminate the search.

8. Otherwise halve λ_{inv}.

3.4 Clustering Technique

After designing a rotationally invariant autoencoder, the final step in rotationally invariant cloud clustering (RICC) is to cluster the latent representations produced by the encoder for a set of cloud images. We apply hierarchical agglomerative clustering (HAC) [63] to merge pairs of clusters from bottom to top, minimizing at each step the linkage distance among merging clusters. We use Ward's method [172], which minimizes the variance of merging clusters, as the linkage metric. We use HAC, which deals with each data as a singleton cluster, rather than K means++ [4], because the HAC initialization strategy of repeatedly merging data points gives more stable results than the K means++ approach of using random starting centroids.

Suppose that we have two clusters C_A and C_B containing data points $X = \{x_{i_1}, \cdots, x_{i_N}\}$ for cluster $i \in \{A, B\}$. HAC with Ward's method computes the linkage distance as follows,

$$d(C_A, C_B) = E(C_A \cup C_B) - E(C_A) - E(C_B), \tag{3.5}$$

where $E(C_i) = \sum_{x \in C_i}(d(x, c_i))$ for the centroid $c_i = \sum_{x \in C_i} \frac{x}{|C_i|}$, denotes the sum of the weighted square distances between all data points in the cluster and the centroid c_i.

3.5 Protocol of Five Criteria

An inherent difficulty in validating unsupervised approaches is the validation of models aside from loss values when there is no ground truth against which to evaluate the outcomes. While a supervised approach involves a perfect ground truth against which output can be compared, an unsupervised learning system produces outputs whose utility must be more creatively evaluated. Therefore, it is necessary to define a series of evaluation protocols to determine whether the resulting cloud classes via RICC are meaningful and useful.

RICC seeks cloud clusters that: 1) are *physically reasonable* (i.e., embody scientifically relevant distinctions); 2) capture information on *spatial distributions*, such as textures, rather than only mean properties; 3) are *separable* (i.e., are cohesive, and separated from other clusters, in latent space); 4) are *rotationally invariant* (i.e., insensitive to image orientation); and 5) are *stable* (i.e., produce similar or identical clusters when different subsets of the data are used). Table 3.2 summarizes these criteria and the quantitative and qualitative tests that we have developed to validate them.

Table 3.2: Proposed five-criteria evaluation protocol. We design several tests to demonstrate quantitative and qualitative evaluation that distinguishes useful from non-useful autoencoders and clustering results.

Criterion	**Test**	**Requirement**
Physically reasonable	Cloud physics	Non-random distribution; median inter-cluster correlation < 0.6
Spatial distribution	Spatial coherence	Spatially coherent clusters
	Smoothing	Low adjusted mutual information (AMI) score
	Scrambling	Low AMI score
Separable	Separable clusters	No crowding structure
Rotationally invariant	Multi-cluster	AMI score closer to 1.0
Stable	Significance of cluster stability	Ratio of Rand Index $G/R \geq 1.01$
	Similarity of clusterings	Higher Adjusted Rand Index (ARI)
	Similarity of intra-cluster textures	Lower weighted average mean square distance
	Clusters capture seasonal cycle	Minimal seasonal texture difference

3.5.1 Criterion 1: Physically Reasonable Clusters

Cloud clusters produced by RICC should be *physically reasonable*, with scientifically relevant distinctions. To this end, we define a test that examines quantitative differences among cloud physics parameters.

Test 1: Cloud physics parameters. We define reasonableness in terms of whether each cluster produced by RICC is associated with distinct distributions (i.e., not being randomly distributed) of four selected retrieved cloud physics parameters from the MOD06 product: cloud optical thickness (COT), cloud top pressure (CTP), cloud top phase (CPI), and cloud effective radius (CER). We select COT and CTP because these two parameters are used for ISCCP cloud classification to evaluate the compatibility between novel types of clouds and the established ISCCP cloud classes [131] CPI and CER are not considered in the nine ISCCP cloud classes, but able to add additional differences in cloud properties in resulting cloud clusters.

The reasonableness test is defined as follows. We first verify that the values for each parameter collected from $\texttt{OC-Patches}_{\texttt{HAC}}$ are not randomly distributed, which would contradict cloud physics. Then, for each parameter, We examine whether different clusters show different distributions. This is based on calculating for each cluster pair the inter-cluster correlation and then computing the median of the resulting inter-cluster correlation coefficients. If for at least one of the four parameters, this median is less than 0.6, a cutoff commonly used to indicate the empirical threshold of no strong correlation [50] in a variety of natural sciences, then We conclude that the clusters do indeed group patches based on physical properties. A median inter-cluster histogram correlation of less than 0.6 suggests that half of the histogram pairs do not perfectly correlate, and thus we declare that the cloud clusters have distinct patterns.

Note that autoencoders are trained only on the MOD02 input radiances, not on four MOD06 physical parameters. Thus, this test determines whether our training and clustering

process is able to embed into the latent representation the distinctions that are recorded by the MOD06 parameters.

3.5.2 Criterion 2: Spatial Distribution

Cloud clusters produced by RICC should capture information on cloud *spatial distributions*. This means that they should not be reproducible by using only mean properties over the target area. To this end, we define three tests.

Test 2.1: Spatial coherence. We qualitatively evaluate whether the clusters produced by an autoencoder demonstrate more spatially coherent assignments than those obtained by clustering patch-mean cloud parameters. When clustering without an autoencoder, we apply HAC to the patch-mean values of COT, CTP, cloud water path (CWP), and CER.

Test 2.2: Smoothing. We examine how the cluster assignments for cloud images change when we alter the spatial resolution of the images via smoothing.

Test 2.3: Scrambling. We examine how cluster assignments scramble pixels in patches so as to remove spatial patterns while preserving the distribution of values.

If the cluster assignments do not change in the Tests 2.2 and 2.3, we conclude that our autoencoder is not learning spatial information, because the encoder generates similar representations when the input images are transformed to remove spatial information. To quantify the similarity of the clustering assignment, we use the following metric:

Adjusted mutual information (AMI) score The AMI score [166] is a metric that adjusts the mutual information (MI) score to account for a chance to measure the extent to which cluster assignments agree for inputs of different spatial resolutions. Given two

clustering assignments U and V, the AMI is computed as:

$$\mathrm{AMI}(U,V) = \frac{\mathrm{MI}(U,V) - \mathbb{E}(\mathrm{MI}(U,V))}{\mathrm{avg}\{H(U),H(V)\} - \mathbb{E}(\mathrm{MI}(U,V))}, \tag{3.6}$$

where $H(\cdot)$ depicts entropy, which formally defines with probability $P(u)$ as

$$H(U) = -\sum_{u\in U} P(u)\log P(u) \tag{3.7}$$

and $\mathrm{MI}(U,V)$ is the mutual information between clustering assignments U and V, as determined by their joint distribution $P(u,v)$ and their respective probabilities $P(u)$ and $P(v)$:

$$\mathrm{MI}(U,V) = \sum_{u\in U}\sum_{v\in V} P(u,v)\log\frac{P(u,v)}{P(u)P(v)}. \tag{3.8}$$

The AMI of two sets of cluster assignments is 1 if the assignments match perfectly, regardless of the assigned labels, and 0 if there is no match. If the AMI score between the clustering results obtained for two different spatial resolutions (test 2.2) and/or spatial positions (test 2.3) is low, we conclude that the trained autoencoder has learned spatial information in its latent representation; if the agreement score is close to 1, we conclude that it has failed to capture spatial patterns and thus is likely instead only encoding the distribution of pixel values in input images.

Smoothing test implementation Listing 3.1 provides the pseudocode for the smoothing test. We use $\texttt{OC-Patches}_{\texttt{HAC}}$ described in Section 3.1, which contains 74 911 patches used for generating clustering centroids. Let k be the kernel size used to convolve the image with a boxcar filter to produce the smoothed version. The smoothing process computes an average value for each pixel region in a patch. For instance, for $k = 2$, it sets each pixel at (i,j), $(i+1,j)$, $(i,j+1)$ and $(i+1,j+1)$, for $i \in \{1,3,\cdots N-1\}$ and $j \in \{1,3,\cdots N-1\}$, where N is the patch size, to be the average value of those pixels, leaving any remaining border

pixels (e.g., if $N = 5$ and $k = 2$, those with $i = 5$ or $j = 5$) unchanged.

Let P_k be the patches obtained when each of $\texttt{OC-Patches}_{\texttt{HAC}}$ is smoothed over $k \times k$ pixels. Then, we encode each smoothed patch in P_k with the trained encoder, giving Z_k as the corresponding set of latent representations, and apply HAC to those latent representations to obtain a set of k clusters C_k. Lower AMI scores then indicate that the autoencoder has mapped different spatial structures within the input images into different latent representations. Finally, we determine agreement between C_1, obtained from the unsmoothed X_{holdout}, and C_k by computing $\text{AMI}(C_1, C_k)$.

```
# Smooth a single image
def smooth(x, k):
    # fix here
    p = Average(x[i:i+k-1,j:j+k-1])
    return p

# Compare cluster assignments for different kernel sizes
for kernel size k in [list of k]:
    # Smooth each patch for kernel size k
    P_k = smooth(x, k) for x in X_holdout
    # Encode each smoothed patch
    Z_k = encode(x) for x in P_k
    # Cluster resulting latent representations
    C_k = Cluster(Z_k, ncluster)
    # Determine agreement between C_1 and C_k
    compute AMI(C_1, C_k)
```

Listing 3.1: Pseudocode for the smoothing protocol. The `smooth` function computes an average over a local window $k \times k$, as depicted in the figure.

Scrambling test implementation Listing 3.2 presents the scrambling test protocol in pseudocode form. Similar to Smoothing Test, we compare the cluster assignments when we scramble the pixels of each image after smoothing patches by applying a random permutation to the smoothed pixels, a process that we repeat (with the same random permutation) for each channel based on different size of kernels.

Again, we use $\texttt{OC-Patches}_{\texttt{HAC}}$ in evaluating the smoothing protocol. For each kernel size, we smooth each image in X_{holdout} by k and then scramble its pixels, giving P_k. We then compute the latent representation on Z_k^{sc} by encoding P_k^{sc} with a trained encoder and cluster the latent representations obtained for the different images to obtain clusters C_k^{sc}. We assess the agreement of C_k and C_k^{sc} by computing $\text{AMI}(C_k, C_k^{sc})$.

```
# Scramble a single image
def scramble(x):
    indicesA = [(1,1), (1,2), ..., (height(x), width(x))]
    indicesB = random_shuffle(indicesA)
    for (i,j),(m,l) in (indicesA,indicesB):
        p[i,j] = x[l,m]
        p_[l,m] = x_[i,j]
    return p

# Compare cluster assignments for different kernel sizes
for kernel size k in [list of k]:
    # Smooth each patch for kernel size k
    P_k = smooth(x, k) for x in X_holdout
    # Scramble each smoothed patch
    P_k_sc = scramble(x, k) for x in P_k
    # Encode each scrambled patch
    Z_k_sc = encode(x) for x in P_k_sc
```

```
# Cluster resulting latent representations
C_k = Cluster(z_k_sc, ncluster)
# Determine agreement between C_1 and C_k
compute AMI(C_1, C_k)
```

Listing 3.2: Pseudocode for the scrambling protocol. The scramble function shuffles the values of randomly selected pixel pairs, as depicted in the code.

3.5.3 Criterion 3: Separable Clusters

Clusters produced by RICC should be *separable* in a way that there are both cohesive in the same cluster in latent space and separated from each other. To evaluate this, we design a test as follows:

Test 3: Spatial organization. We use t-distributed Stochastic Neighbor Embedding (t-SNE) [87] to examine the spatial organization of the latent representation when projected onto a two-dimensional map, and thus to determine whether similar (dissimilar) clusters, as defined by the physical parameter distributions, are projected to closer (more distant) locations in the embedding space. This test is a qualitative metric to investigate the structure of latent space.

t-distributed Stochastic Neighbor Embedding (t-SNE) t-SNE is a probabilistic nonlinear dimensionality reduction technique that maps each data point in a high-dimensional space to a lower-dimensional point in such a way that, with high probability, similar data are placed near to each other and dissimilar data far apart. Suppose that we have N latent representations $Z = \{z_1, \cdots, z_i, z_j, \cdots, z_N\}$ produced by an autoencoder. Let P be a joint probability distribution in the high-dimensional input space, and Q a joint probability distribution in the low-dimensional projection space. The t-SNE optimization minimizes the Kullback-Leibler (KL) divergence between $p_{j|i}$ and $q_{j|i}$. The conditional probability $p_{j|i}$ for

the M-dimensional latent representation produced by our autoencoders is:

$$p_{j|i} = \frac{\exp(-\|z_i - z_j\|^2/2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|z_i - z_k\|^2/2\sigma_i^2)}, \tag{3.9}$$

where σ_i denotes the variance of a Gaussian distribution for data point z_i, while for a two-dimensional map:

$$q_{j|i} = \frac{\left(1 + \|z'_i - z'_j\|^2\right)^{-1}}{\sum_{k \neq l} \left(1 + \|z'_l - z'_k\|^2\right)^{-1}}. \tag{3.10}$$

The cost function to be minimized by gradient descent is then

$$\mathrm{KL}(P||Q) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}. \tag{3.11}$$

Note that this joint distribution assumes a Student t-distribution with one degree of freedom in the low-dimensional map.

3.5.4 Criterion 4: Rotation Invariance

We expect the classes produced by RICC to be *rotationally invariant*, where HAC assigns a cloud image and its rotated version into the same cluster regardless of its orientation regardless of the orientation of the image in which they appear.

Test 4: Multi-cluster rotation-invariance. We evaluate whether our RI autoencoder groups similar types of clouds into the same cluster regardless of image orientation.

Multi-cluster rotation-invariance We use `Test` dataset, a set of 2000 patches that are randomly sampled from `OC-Patches` as well as not considered during training as holdout patches. We make 11 copies of each patch in this set such that we rotate each copy every 30°; thus, the ideal result should return the same cluster label for both the original patches and the rotated copies. We then implement HAC clustering from 4 to 2000 clusters. The AMI

score should be close to 1 for 2000 clusters because each patch among 2000 `OC-Patches`and its replications can be placed in a unique cluster.

3.5.5 Criterion 5: Stable Clusters

Cluster stability is an important property for a cloud classification algorithm. A clustering method is said to be *stable* for a dataset, D, and number of clusters, k, if it produces similar or identical cluster assignments when applied to different subsets of D. To evaluate stability of clusters, we define four tests to evaluate this criterion:

Test 5.1: Clustering similarity. We measure *clustering similarity* by generating clusterings for different subsets of the same dataset, and calculating the average distance between those clusterings.

Test 5.2: Clustering similarity significance. We measure *clustering similarity significance* by comparing each clustering similarity score to that obtained when our clustering method is applied to data from a uniform random distribution.

Test 5.3: Intra-cluster texture similarity. We measure *intra-cluster texture similarity* by calculating the average distance between latent representations in each cluster.

Test 5.4: Seasonal stability. We measure *seasonal stability* by comparing intra-cluster texture similarity for patches from January and July.

We are concerned not only with determining whether RICC generates clusters that are stable, but also with identifying the optimal number of clusters, k^*. In determining that number, we must consider all four tests just listed: we want a high clustering similarity, a high significance (certainly greater than 1), a low intra-cluster similarity score, and low intra-seasonal texture differences.

For all of our stability tests, we work with $D = \{$ `OC-Patches` from 2003 to 2021, inclusive $\}$. $|D| \approx 180M$. (We do not consider data from 2000–2002 because Aqua and Terra were not operating simultaneously for an entire year-long observation during that period.) We create a holdout subset H with number of patches $N_H = 14\,000$, and create 30 random subsets S_i

with $N_R = 56\,000$ by sampling without replacement from $D \setminus H$. This procedure ensures that the different S_i are mutually exclusive and that there is no intersection between our holdout set H and the random subsets. The ratio $N_H : N_R$ of 20 : 80 is standard practice. We then create our 30 test datasets as $H \cup S_i$ for $\forall i \in \{1, \cdots, 30\}$. To quantify the similarity of clustering assignments, we use the following metrics:

Rand Index The Rand index [125] measures the similarity of two partitions of clustering assignments. Let $U = \{U_1, \cdots, U_r\}$ and $V = \{V_1, \cdots, V_c\}$ be two clustering partitions of a set of N objects $O = \{o_1, \cdots, o_{N_P}\}$, such that $\bigcup_{i=1}^{r} U_i = \bigcup_{j=1}^{c} V_j = O$, and $U_i \cup U_{i'} = \emptyset$ as well as $V_j \cup V_{j'} = \emptyset$ for $1 \leq i \leq r$ and $1 \leq j \leq c$. We count how many of the $\binom{N}{2}$ possible pairings of elements in O are in the same or different clusters in U and V:

- P_{11}: number of element pairs that are in the *same* clusters in both U and V
- P_{10}: number of element pairs that are in *different* clusters in U, but in the *same* cluster in V
- P_{01}: number of element pairs that are in the *same* cluster in U, but in *different* clusters in V; and
- P_{00}: number of element pairs that are in *different* clusters in both U and V.

The Rand index then computes the fraction of correct cluster assignments:

$$\text{RandI}(U, V) = \frac{P_{11} + P_{00}}{P_{11} + P_{10} + P_{01} + P_{00}} = \frac{P_{11} + P_{00}}{\binom{N}{2}} \tag{3.12}$$

It has value 1 if all pairs of labels are grouped correctly and 0 if none are correct. The metric is independent of the absolute values of the labels: that is, it allows for permutations.

To illustate how the Rand index works, consider the two clusterings: $A = \{d_1\}, \{d_2, d_3\}$ and $B = \{d_1, d_2\}, \{d_3\}$ of the dataset $D = \{d_1, d_2, d_3\}$. Here, $N = 3$, and there are $\binom{3}{2} = 3$ possible pairings of the three dataset elements: $(d_1, d_2), (d_1, d_3), (d_2, d_3)$. Thus: $P_{11} = 0$,

as no pair is in the same cluster in both A and B; $P_{10} = 1$, as d_1 and d_2 are in different clusters in A but the same cluster in B; $P_{01} = 1$, as d_2 and d_3 are in different clusters in A but the same cluster in B; and $P_{00} = 1$, as d_1 and d_3 are in different clusters in both A and B. Hence, the Rand index by Equation 3.12 of A and B is $(0 + 1)/3 = 0.33$.

Adjusted Rand Index A difficulty with the Rand index is that its value tends to increase with the number of clusters, hindering comparisons across different numbers of clusters. In order to permit comparisons of Rand index values across different numbers of clusters, the **adjusted Rand index** (ARI) [53] corrects for co-occurrences due to chance:

$$\mathrm{ARI}(U, V) = \frac{\binom{N}{2}(P_{11} + P_{00}) - [(P_{11} + P_{10})(P_{11} + P_{01}) + (P_{01} + P_{00})(P_{10} + P_{00})]}{\binom{N}{2}^2 - [(P_{11} + P_{10})(P_{11} + P_{01}) + (P_{01} + P_{00})(P_{10} + P_{00})]}, \tag{3.13}$$

where the P_{xy} are as defined above.

Stability Test 5.1: Clustering Similarity We measure clustering similarity by first generating clusterings for different subsets of the target dataset and then calculating the average pairwise distance between those clusterings. This approach is documented as Algorithm 1. As described above, we use as the input dataset D all ocean-cloud patches from 2003–2021, inclusive. We then define a holdout set, H, $H \cup S_i$, $i \in 1..30$, to generate 30 different clustering assignments via a trained RICC for evaluation (line 1), and use as a set of "perturbed versions" N subsets selected without replacement from $D \setminus H$ (line 3). Then for each number of clusters, k, in the range $8 \leq k \leq k_{\max}$, I then: train RICC on each subset (line 8); apply the trained RICC to generate a clustering for the holdout set (line 6); use the adjusted Rand index, ARI, to evaluate pairwise distances between those clusterings (line 10); and average among the 30 clusterings generated by the RICC models $\{\mathrm{RICC}_k^i$, $i \in 1..30\}$ to determine the mean clustering similarity for that specific cluster number k. Finally, we calculate the ARI for all $\binom{30}{2} = 435$ combinations of those 30 clusterings and determine the mean ARI score $G_8..G_{k_{max}}$ (line 12).

Algorithm 1 Pseudocode for the clustering similarity test described in Section 3.5.5.

Input: D: { `OC-Patches` for 2003–2021, inclusive }
Output: $G_8, \cdots, G_{k_{\max}}$: Clustering similarity scores for cluster counts from 8 to k_{max}.

1: $H := \{x \mid x \in D\}$ where $|H| = N_H$ ▷ Select holdout set to be used for evaluation
2: **for** i from 1 to N **do**
3: Select a subset $S_i := \left\{x \mid x \in D \setminus H \setminus \bigcup_{j=1}^{i-1} S_j\right\}$ with $|S_i| = N_R$
4: **for** k from 8 to $k_{\max}$ **do**
5: $\texttt{RICC}_k^i \leftarrow$ Train RICC with k clusters on $S_i \cup H$
6: $C_k^i \leftarrow \texttt{RICC}_k^i(H)$ ▷ Determine cluster assignments in H with $\texttt{RICC}_k^i$
7: **end for**
8: **end for**
9: **for** k from 8 to $k_{\max}$ **do**
10: $G_k = \frac{1}{\binom{N}{2}} \sum_{(i,j)\in\binom{N}{2}} \texttt{ARI}\left(C_k^i,\ C_k^j\right)$ ▷ Mean similarities for RICC clusters
11: **end for**
12: Return clustering similarity scores $\{G_8, \cdots, G_{k_{max}}\}$

Stability Test 5.2: Significance of Similarities Having determined how cluster similarity scores vary with a number of clusters, we next turn to the question of whether these values are significant. Following Von Luxburg [168], we compare cluster similarity scores, as shown in Algorithm 2, against those obtained when the same method is applied to data generated not by our trained autoencoder but from a random uniform distribution clustered with the same HAC method. To produce random label assignments, we first prepare 30 datasets that are sampled from random uniform distributions $\mathcal{U} \in [-2\sigma, 2\sigma]$ (line 6). For each k in the range $8..k_{\max}$, We apply HAC to the random data to generate random labels (line 11), from which we also calculate the Rand Index for 435 combinations, giving the mean scores $R_8..R_{k_{max}}$ (line 17). Finally, we compare how the ratio $\frac{G_k}{R_k}$ between those two values varies with number of clusters, k (line 20). We then compute the mean clustering similarity score G from our patches and R from the data from the random uniform distribution for each k for all 435 combinations, though here we use the Rand index rather than ARI, as we are not comparing scores across k. We can then compare how the ratio between those two values varies with number of clusters. A ratio > 1 indicates that cluster assignments are more stably grouped than would be expected by chance; a value of 1 indicates that there is no

benefit to adding extra clusters.

Algorithm 2 Pseudocode for the stability significance test described in Section 3.5.5.

Input: D: { `OC-Patches` for 2003–2021, inclusive }, trained rotationally invariant autoencoder AE

Output: $\{\frac{G_8}{R_8}, \cdots, \frac{G_{k_{\max}}}{R_{k_{\max}}}\}$: cluster similarity significance scores

$H := \{x \mid x \in D\}$ where $|H| = N_H$ ▷ Select holdout set to be used for evaluation

$z = \{AE(x) : x \in H\}$ ▷ Use trained autoencoder to compute latent representations

$\sigma = \sqrt{\frac{1}{N_H} \sum_{j=1}^{N_H} (z_j - \overline{z})^2}$ ▷ Calculate standard deviation σ for latent representations

for i from 1 to N **do**

  Select a subset $S_i := \{x \mid x \in D \setminus H \setminus \bigcup_{j=1}^{i-1} S_j\}$ with $|S_i| = N_R$

  Sample $U_i := \{u \mid u \in \mathcal{U}[-2\sigma, 2\sigma]\}$ with $|U_i| = N_H$, $\mathcal{U}$ a random uniform distribution.

  for k from 8 to $k_{\max}$ **do**

    $\texttt{RICC}_k^i \leftarrow$ Train RICC on $S_i \cup H$

    $\texttt{RICC}_k^i(H) \leftarrow$ Determine cluster assignments in H

    $\texttt{HAC}_k^i \leftarrow$ Train HAC on U_i

    $\texttt{HAC}_k^i(U_i) \leftarrow$ Determine cluster assignments in U_i

  end for

end for

▷ Calculate averages of cluster similarities, as computed via Rand index, $\texttt{RandI}(\cdot)$, between all pairs of two label assignments resulting from RICC and a random distribution respectively

for k from 8 to $k_{\max}$ **do**

$$G_k = \frac{1}{\binom{N}{2}} \sum_{(i,j) \in \binom{N}{2}} \left[\texttt{RandI}\left(\texttt{RICC}_k^i(H),\ \texttt{RICC}_k^j(H)\right)\right]$$ ▷ Mean similarities for RICC clusters

$$R_k = \frac{1}{\binom{N}{2}} \sum_{(i,j) \in \binom{N}{2}} \left[\texttt{RandI}\left(\texttt{HAC}_k^i(U_i),\ \texttt{HAC}_k^j(U_j)\right)\right]$$ ▷ Mean similarities for random clusters

  Calculate $\frac{G_k}{R_k}$, ratio of stability between RICC and random samples

end for

Return cluster similarities significance scores, $\{\frac{G_8}{R_8}, \cdots, \frac{G_{k_{\max}}}{R_{k_{\max}}}\}$

Stability Test 5.3: Intra-cluster Texture Similarity A stable clustering should group patches with similar textures within the same cluster. To determine whether a clustering has this property, we examine how the average distance between latent representations within each cluster changes when we apply RICC to create different numbers of clusters. The mean distance between pairs of latent representations in a cluster relates to similarity of texture, as our RI autoencoder learns texture features and encodes those features in latent representations. Specifically, we calculate the mean squared Euclidean distance between the

latent representations computed for patches in our holdout set H.

For a clustering with k clusters, let n_c be the number of elements in cluster c, and y_1 .. y_{n_c} be the patches in that cluster. As cluster sizes can vary, we weight each cluster's mean distance by $w_c = n_c / \sum_{i=1}^{k} n_i$, to obtain a weighted average mean squared Euclidean distance:

$$d_k = \sum_{c=1}^{k} \left(w_c \sum_{i=1}^{m} \sum_{j>i}^{m} \frac{||z(y_i) - z(y_j)||_2^2}{\frac{m}{2}(m-1)} \right) \quad \text{where } m = \min(n_c, N_p), \tag{3.14}$$

where z represents the latent representations generated by our RI autoencoder, and N_p is the maximum number of patches to consider in the distance calculation—a limitation used to accelerate calculations. We set $N_p = 200$ for our tests. Note that when the total number of clusters is large, some individual clusters may have a size less than this limit.

We calculate Equation 3.14 for k from 8 to 256 for each of our 30 clusterings of test subsets $\{\text{RICC}_k^1(H), \cdots, \text{RICC}_k^{30}(H)\}$, and then compute the mean value across clusterings. The resultant weighted average distance decreases monotonically with the cluster number k: see Figure 3.11c), as does the metric G/R from test 2, but the trends have opposite implications: lower values are worse in test 2 but better in test 3. A lower distance value indicates that cloud texture and physical properties are more homogeneous within a given cluster, meaning the resultant set of k cloud classes provides a more consistent cloud diagnostic. The implication is that the optimal number of clusters k^* will be approximately the largest number that satisfies our criterion in test 5.2.

Stability Test 5.4: Seasonal Variation of Textures within Clusters Our final test investigates whether clusters produced via RICC show similar patterns regardless of season: we compare intra-cluster texture similarity between `OC-Patches` from January and July. If differences are small, the number of clusters used is sufficient to accommodate the large seasonal changes in cloud morphology.

We use RICC with the autoencoder trained on $\texttt{OC-Patches}_{\texttt{AE}}$ and cluster centroids based

on $\texttt{OC-Patches}_{\texttt{HAC}}$, for different numbers of clusters k, as before. For each k, we then apply the trained RICC_k model to the patches in $\texttt{OC-Patches}_{\texttt{HAC}}$ to assign a label $c \in \{1, .., k\}$ to each patch, and for each c, extract the latent representations for m_c^s randomly selected July patches and m_c^w randomly selected January patches with that label (with m_c^s and m_c^w being at most 100 in these analyses, but less if a particular cluster has fewer January or July patches, respectively), compute an intra-cluster texture similarity score for each set of July and January patches, and (as in Section 3.5.5) weight each cluster mean by the actual m_c^s or m_c^w so that we can consider texture similarities from many clusters without results being dominated by trivial clusters that we observe to group fewer similar patches due to undersampling. We then sum the scores to obtain the overall weighted averaged squared distance (WASD) for k clusters. In summary:

$$\text{WASD}_k = \sum_{c=1}^{k} \left(w_c \sum_{i=1}^{m_c^s} \sum_{j=1}^{m_c^w} \frac{||z(y_i^s) - z(y_j^w)||_2^2}{m_c^s \cdot m_c^w} \right) \tag{3.15}$$

where w_c and z are as defined in Section 3.5.5 and $y^s = \{y_1^s \; .. \; y_{m_c^s}^s\}$ and $y^w = \{y_1^w \; .. \; y_{m_c^w}^w\}$ are the January and July patches in cluster c, respectively.

We expand the analysis to account for two additional potential sources of bias. Because the specific days used in $\texttt{OC-Patches}_{\texttt{HAC}}$ may affect our results, we assemble two additional versions of $\texttt{OC-Patches}_{\texttt{HAC}}$, selecting two days without replacement from each season in 2003, as before. The resulting $\texttt{OC-Patches}_{\texttt{HAC-2}}$ and $\texttt{OC-Patches}_{\texttt{HAC-3}}$ have 77 235 and 76 143 patches, respectively. Similarly, to account for any effect of the random selection of the m^s summer and m^w winter patches, we repeat the analysis of Equation 3.15 three times for each of $\texttt{OC-Patches}_{\texttt{HAC}}$, $\texttt{OC-Patches}_{\texttt{HAC-2}}$, and $\texttt{OC-Patches}_{\texttt{HAC-3}}$. In this way we obtain a total of $9 \cdot k$ mean squared distance values and nine WASD values for each k in the range 8 to 256.

We first determine the optimal combination of coefficients in Equation 3.2, investigate the performance of RI autoencoder in terms of their robustness (i.e., to what extent clustering assignment changes among autoencoders trained in different initialization), and scaling as a

function of GPUs. Then, we examine clusters based on our evaluation protocol (see Section 3.5).

3.6 Determine optimal combination of λ

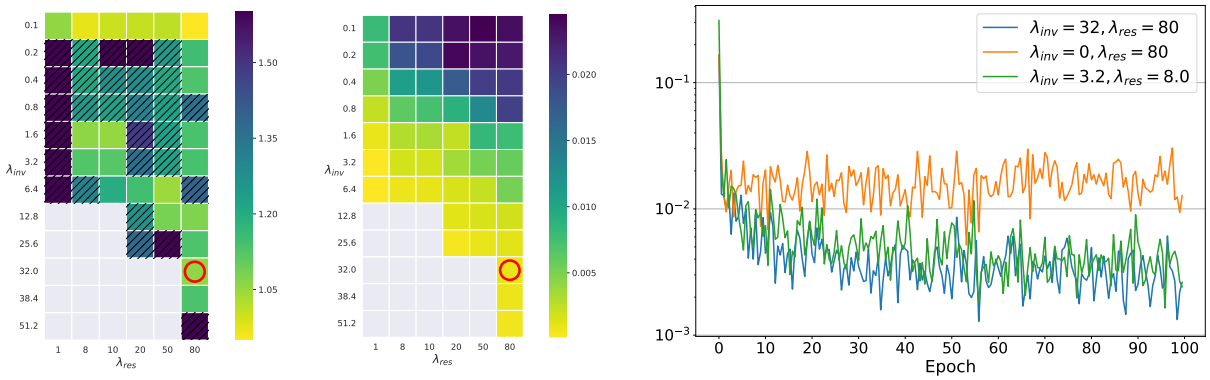
We first implement a grid search to determine the optimal combination of weight parameters $(\lambda_{\text{inv}}, \lambda_{\text{res}})$ based on the grid search protocol (see Section 3.3) because the performance of RI autoencoder is sensitive to the values. Figure 3.3 shows that the optimal parameter combination is $(\lambda_{\text{inv}}, \lambda_{\text{res}}) = (32, 80)$ for learning rate at 10^{-2}. Other parameter combinations do not satisfy the restoration error ratio criterion $[L_{\text{res}}(\mathbf{B}, X_{\text{holdout}}) > 1.2L_{\text{res}}(\mathbf{A}, X_{\text{holdout}})]$ (i.e., a newly trained model B for larger λ_{inv} produces restoration errors 20% larger than the baseline model A) and/or do not achieve rotation-invariant (i.e., a set of the restored image $\mathbf{B}(\mathcal{R}(x_i, \theta))$ for $\theta \in \{0, 30, \cdots, 330\}$ is not identical for all rotation combinations.)

Figure 3.3a shows the ratio of restoration loss $L_{\text{res}}(\mathbf{B}, X_{\text{holdout}})/L_{\text{res}}(\mathbf{A}, X_{\text{holdout}})$. Hatched elements represent suboptimal combinations by $L_{\text{res}}(\mathbf{B}, X_{\text{holdout}}) > 1.2L_{\text{res}}(\mathbf{A}, X_{\text{holdout}})$. There are more suboptimal elements colored in navy seen at lower λ_{res} ($\lambda_{\text{res}} = 1, 10, 20$) and larger λ_{inv} (e.g., $(\lambda_{\text{inv}}, \lambda_{\text{res}}) = (25.6, 50), (51.2, 80)$), suggesting that a larger λ_{res} help to decrease the restoration error ratio and allow to take larger λ_{inv} to satisfy the criterion.

Figure 3.3b shows the standard deviation of the cosine similarities for RI autoencoder outputs as a function of λ values, which enables quantitative investigation of rotation-invariance. Cosine similarity quantifies the similarity between two vectors X and X' in terms of the cosine of the angle between them:

$$\text{Cosine similarity} = \frac{< X, X' >}{\|X\| \cdot \|X'\|}. \tag{3.16}$$

Equation (3.16) gives 1 when elements of two vectors are exactly matched. A standard deviation closer to 0 in Fig. 3.3b means that cosine similarities in restored images rotated for $\theta \in \{0, 30, \cdots, 330\}$ obtain an identical representation, which verifies the autoencoder maps images with different transformations into a single canonical orientation. The matrix



(a) Ratio of restoration losses

(b) Similarity of restored images.

(c) Training losses for different λ combinations.

Figure 3.3: Results of grid search for RI autoencoder. (a) Ratio of the two restoration losses $L_{\text{res}}(\mathbf{B}, X_{\text{holdout}})/L_{\text{res}}(\mathbf{A}, X_{\text{holdout}})$ for different λ combinations. (b) Standard deviation of cosine similarity computed on the restoration images from RI autoencoder, when feeding various transformations rotated by $\{0°, 30°, \cdots, 330°\}$. We highlighted the optimal combination by a red circle. (c) Training losses for the optimal λ combination (blue) and two suboptimal combinations (orange and green) examined in the grid search.

of the ratio of restoration and standard deviations tells that increasing λ_{res} needs increasing λ_{inv} to satisfy the rotation-invariant criterion while increasing λ_{res} leads to better-quality restorations.

Fig. 3.3c shows the evolution of training loss over iterations with different parameter values. The lowest convergence of the blue line in Fig. 3.3c proves that the optimal combination $(\lambda_{\text{inv}}, \lambda_{\text{res}}) = (32, 80)$ achieves the lowest rotation-invariant loss of the combinations shown. Note that an appropriate size of learning rate is also essential to discovering the optimal configuration. Learning rates lower than the 10^{-2} enables the network to have larger $(\lambda_{\text{inv}}, \lambda_{\text{res}})$ wheres considered here allow the network to take larger λ values, which results in the latent representation not being able to achieve rotation-invariance.

3.7 Performance of RI autoencoder

First we examine the robustness of RI autoencoders in terms of clustering assignments if multiple RI autoencoders, which have an identical architecture and training dataset but

are trained on different initialization, generate to what extent similar cluster assignments based on clustering dataset $\texttt{OC-Patches}_{\texttt{HAC}}$, giving robustness of autoencoder. We test six different RI autoencoders from 8 to 256 clusters. RI autoencoder from Kurihana et al. [76] is the baseline and we train five other models: one is RI autoencoder that is trained until 400 epochs (hereafter, 2021–400epochs) because longer training epochs benefit models to gain more generalized representations [19]. The other four models (Model–A – Model–D) are trained based on different initialization for 400 epochs. Figure 3.4 plots the results of the robustness of clustering assignments via the Rand score index as a function of the number of clusters. The Rand index score curve starts from 0.83 at 8 clusters, and converges to 0.99 at 256 clusters, suggesting that RI autoencoders can generate almost identical latent representations, resulting in similar cluster groups via HAC.

Next, we investigate the scaling performance of the RI autoencoder when training models with distributing patches in data parallelism. There are two scenarios in the scaling experiment: 'Strong' scaling measures the changes in an execution time as a function of the number of processors when the total problem size remains, and 'weak' scaling measures the changes in an execution time as a function of the number of processors when a problem size is constant per processor. Particularly in deep learning training, strong scaling lets the total batch size a constant, whereas weak scaling lets the per-GPU batch size a constant and both experiments vary the number of GPUs to train a neural network. Table 3.3 summarizes the number of GPUs used for training, the size of minibatch per GPU, and the total size of GPUs.

Table 3.3: Configuration of strong and weak scaling experiment for the RI autoencoder.

Experiment	Strong scaling								Weak scaling							
GPUs	1	2	4	8	16	32	64	128	1	2	4	8	16	32	64	128
#batch per GPU	512	256	128	64	32	16	8	4	128	128	128	128	128	128	128	128
Total #batch	512	512	512	512	512	512	512	512	128	256	512	1024	2048	4096	8192	16384

Figure 3.5 plots the results of strong and weak scaling based on a training time per step and throughput from weak scaling. Figure 3.5b indicates that the training of RI autoencoder

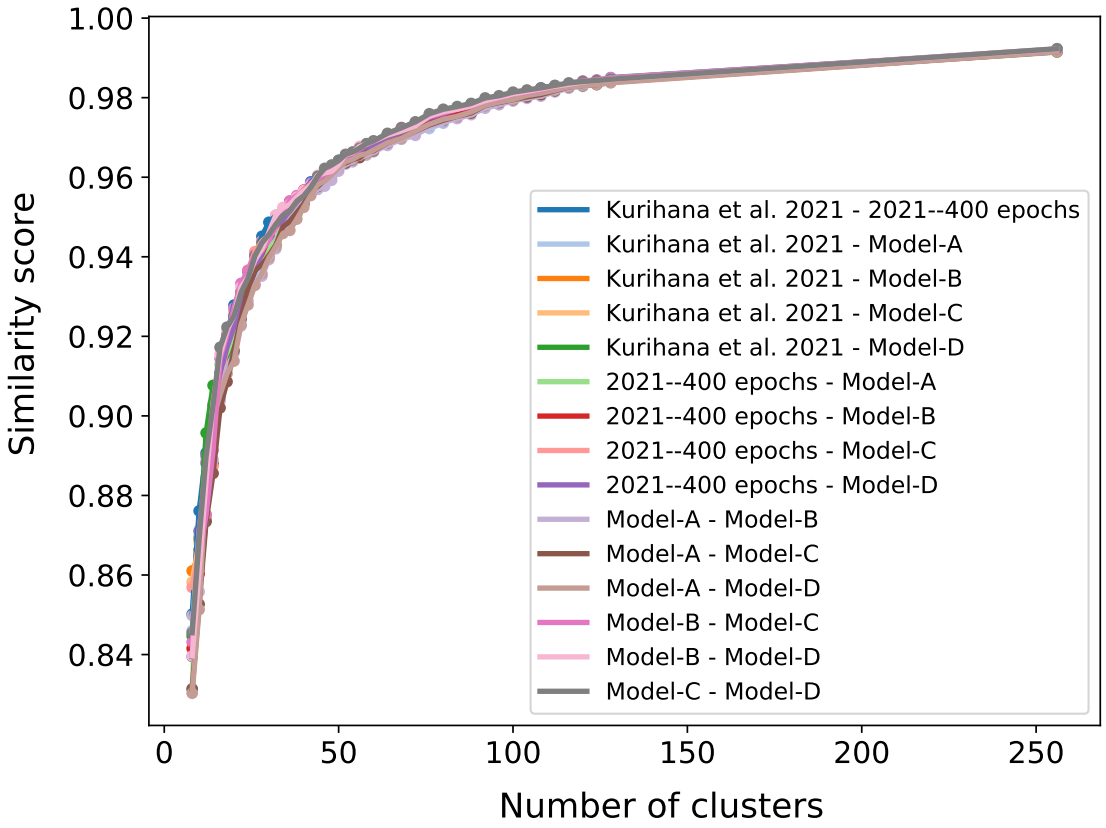
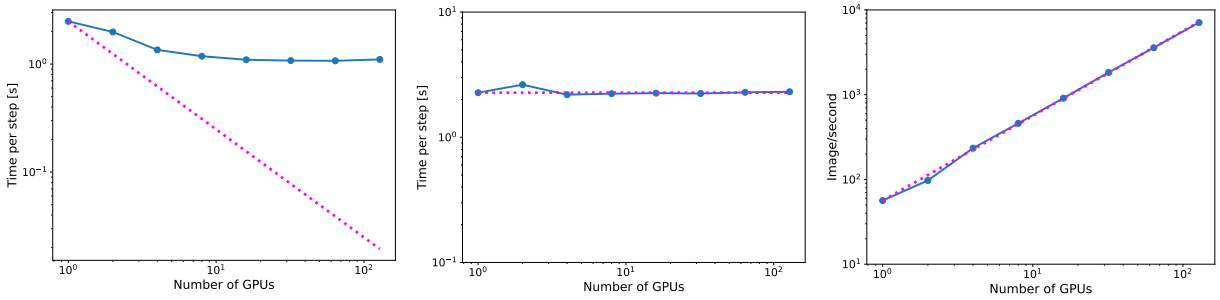


Figure 3.4: Plots of clustering agreement score (Rand Score Index) among six RI autoencoders based on $\texttt{OC-Patches}_{\texttt{HAC}}$ from 8 to 256 clusters. Agreement scores show similar ranges among models for the number of clusters tested, indicating that RI autoencoders are robust to generate similar latent representation and thus result in similar clustering assignments in particular for a larger number of clusters.

efficiently leverages a larger size of GPUs under a weak scaling experiment by 128 GPUs, and the throughput line in Figure 3.5c matches with the ideal scaling curve. Instead, Figure 3.5a suffers from effective scaling in data parallel training, suggesting that the effective weak scaling achieves easier than strong scaling for the proposed RI autoencoder with increasing size of GPUs.

3.8 Evaluation

We evaluate the performance of RICC in terms of the evaluation protocol described in Section 3.5 and compare results obtained from the non-rotationally invariant (NRI)



(a) Strong scaling (b) Weak scaling (c) Throughput in weak scaling

Figure 3.5: Plots of strong and weak scaling of RI autoencoder. (a) Strong scaling and (b) weak scaling results in terms of execution time per step. (c) Throughput of images per second based on weak scaling. The blue line and dots represent scaling performance and the dashed line (magenta) represents the ideal scaling line. Convergence is efficiently scaled in weak scaling but strong scaling suffers from effective scaling.

autoencoder.

3.8.1 Physically Reasonable

Figure 3.6 shows, for each of k clusters from 8 to 256 for the $\texttt{OC-Patches}_{\texttt{HAC}}$ dataset, the median inter-cluster correlation as a quantitative metric to evaluate the distinct combinations of physical properties grouped by each cluster. We set 0.6 as a cut-off threshold following the standard practice to judge whether two datasets are not strongly correlated. Liquid and ice phase (CPI), cloud top pressure, and cloud effective radius are below our cut-off threshold of 0.6, particularly from 16 to 46, and larger than 128 clusters. Cloud optical thickness (COT) is the highest median correlation of 0.87 at 8 clusters and still shows a high correlation throughout the experiment. The diversity of distributions of physical parameters between 16 and 46 can conclude that increasing the number of clusters benefits the separation of fine differences identified in RICC. We observe that CTP and CER are above the cut-off between 48 to 88 clusters, indicating that the relatively larger cluster numbers may have clusters that are indistinct in terms of their physical properties.

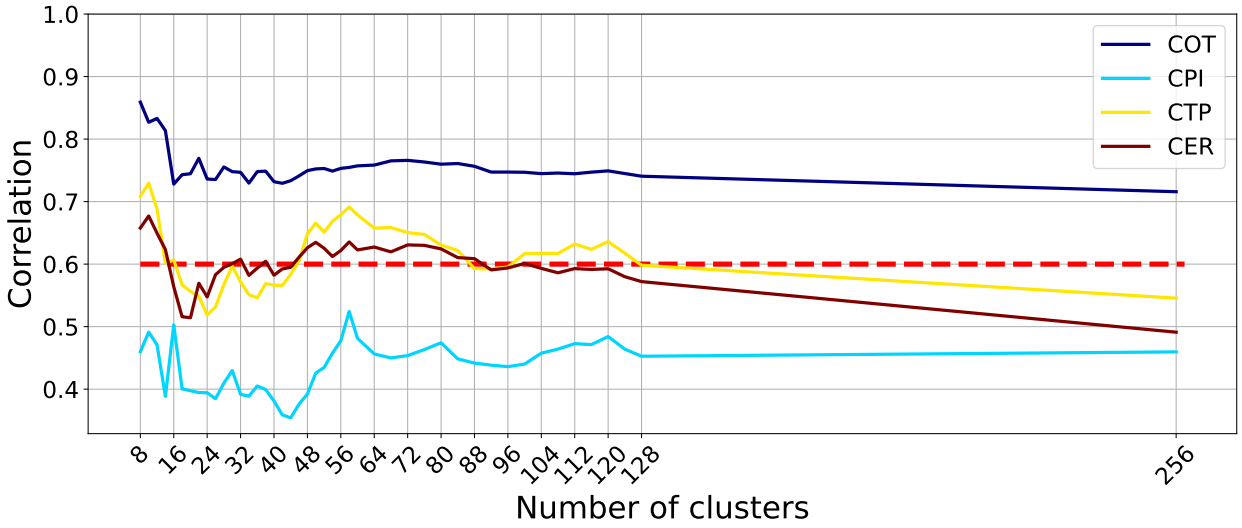


Figure 3.6: Plots of the median of correlation coefficients for four cloud physics parameters (COT, CPI, CTP, and CER) for RI autoencoders. Correlation coefficients <0.6 is a threshold for whether resulting clusters can group unique distributions of physical parameters. Three of the four variables fall below 0.6 from 16 to 46 and > 128 clusters, whereas the majority of clusters have highly similar optical thickness distributions.

3.8.2 Spatial Distribution

First, we conduct a qualitative test to compare whether the autoencoder-based approach outperforms a simple clustering of cloud parameters in terms of their spatial coherence in cluster assignments. Results shown in Figure 3.7 indicate that autoencoder helps to group adjacent cloud patches into the same clusters, resulting in more spatially coherent cluster assignments than cluster assignments produced from HAC applied to sets of four patch-mean cloud parameters.

Next, we apply two different pixel-based sampling processes 'smoothing' and 'scrambling' to patches to investigate to what extent autoencoders embed spatial information into the latent representations.

The smoothing test examines how cluster assignments change when we alter the spatial resolution of patches in $\texttt{OC-Patches}_{\texttt{HAC}}$ via smoothing based on our protocol Section 3.5.2. Lower AMI scores indicate that latent representations vary based on differences in the spatial structure of patches. Figure 3.8a compares the results obtained from RI autoencoder and

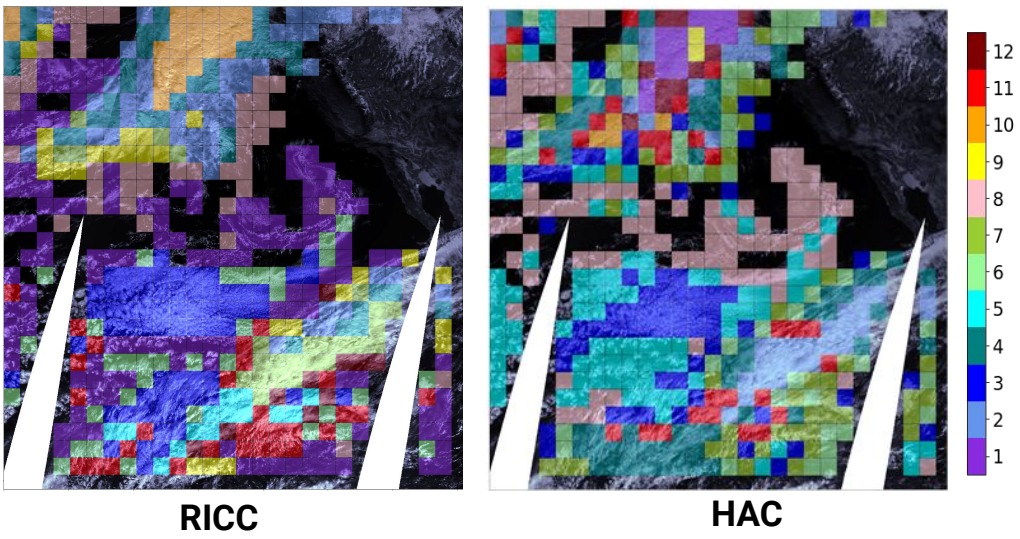
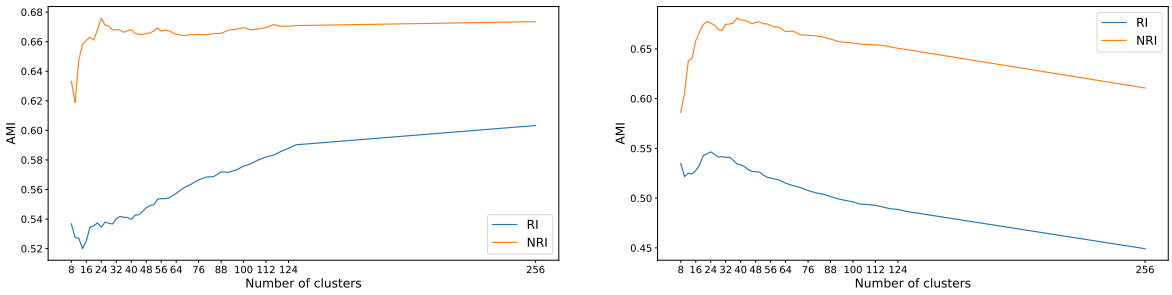


Figure 3.7: On the same MODIS swath with 493 patches, we show; Left: Clusters produced by RICC; Right: Clusters produced by HAC applied to patch-mean values of COT, CTP, CWP, and CER. The background raw visible image from MODIS band 1 is provided to show the context of geolocation and the presence of clouds. For both cases, 12 clusters (shown in the color bar) are applied to examine the spatial distribution of clusters qualitatively. RICC can produce spatially more coherent cluster assignments, whereas a simple clustering to cloud parameter falls short to capture spatial information.

NRI autoencoder from 8 to 256 clusters. Results tell that RI autoencoder achieves the lowest agreement score of 0.519, while NRI obtains 0.619. AMI scores from RI autoencoder are always below the scores from NRI autoencoder.

The scrambling test examines how cluster assignments change when we scramble pixels on patches from $\texttt{OC-Patches}_{\texttt{HAC}}$ to remove spatial patterns while preserving the distribution of pixel values. Similar to the smoothing test, a low agreement score indicates that the trained autoencoder is encoding information about spatial patterns in the latent representation, and a high agreement score shows that it is not. Figure 3.8b shows that RI autoencoder achieves the lowest AMI score of 0.449 and even the highest AMI score of 0.54 is lower than the lowest AMI score of 0.586 from NRI autoencoder.

The results suggest that both autoencoders are learning spatial patterns as their AMI score is not close to 1, a perfect matching of two clustering agreements, and the difference



(a) Smoothing (b) Scrambling

Figure 3.8: AMI scores for kernel size 12 from 8 to 256 clusters based on (a) smoothing test and (b) scrambling test. Lower AMI scores indicate better performance in this test as latent representations result in differently with and without smoothing and scrambling operations. The blue line represents adjusted mutual information between clustering assignments from RI autoencoder with and without smoothing and/or scrambling operation, and the orange line represents those from NRI autoencoder.

can be accounted for with and without rotation-invariance.

3.8.3 Separable Clusters

The separable clusters test qualitatively examines whether patches from the same cluster group are located close to each other and whether those from different cluster groups are far apart from each other in the high-dimensional space. We apply t-SNE to the latent representations from RI autoencoder so as to visualize them in two-dimensional space. Figure 3.9 shows the structure of latent representations of $\texttt{OC-Patches}_{\texttt{HAC}}$, colored by five main cloud groups: *High* altitude, *Medium* altitude, *Sparse*, *Open* cell stratocumulus, and *Closed* cell stratocumulus clouds based on texture and within-cluster mean cloud optical thickness and cloud top pressure values. This is for generalizing the relationship between positions of patches and cluster labels in various cluster numbers k from 8 to 256. We observe that the cloud groups are locally homogeneous and distinct instead of being randomly distributed, indicating that clustering achieves good separability.

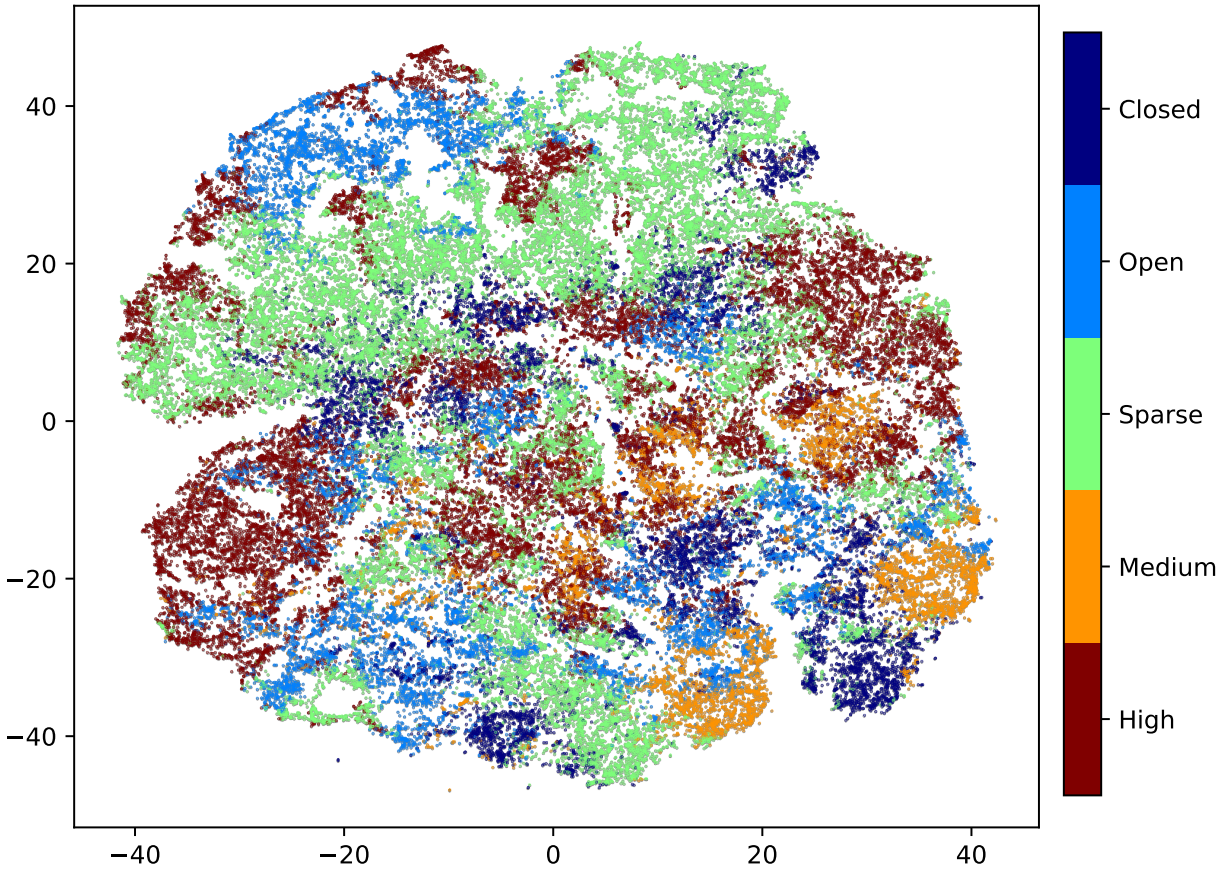


Figure 3.9: t-SNE visualization of latent representations of $\texttt{OC-Patches}_{\texttt{HAC}}$. To generalize the relationship between positions of patches and cluster labels in various cluster numbers k from 8 to 256, we regroup a set of clusters into *High* altitude, *Medium* altitude, *Sparse*, *Open* cell stratocumulus, and *Closed* cell stratocumulus clouds based on texture and within-cluster mean cloud optical thickness and cloud top pressure values.

3.8.4 Rotation Invariance

Fig. 3.10 plots AMI scores as a function of the number of clusters. The AMI curve for the RICC (blue) converges to 0.9, indicating that the clustering result is agnostic to the orientation of clouds in the holdout set. In contrast, the agreement score for the combination of NRI autoencoder and HAC decreases as the number of clusters increases, suggesting that the NRI autoencoder's latent representation is influenced by the rotation of images even if they are for the same types of clouds. Therefore, we conclude that RICC is functional to process a real image dataset when input orientation does not matter for pattern recognition.

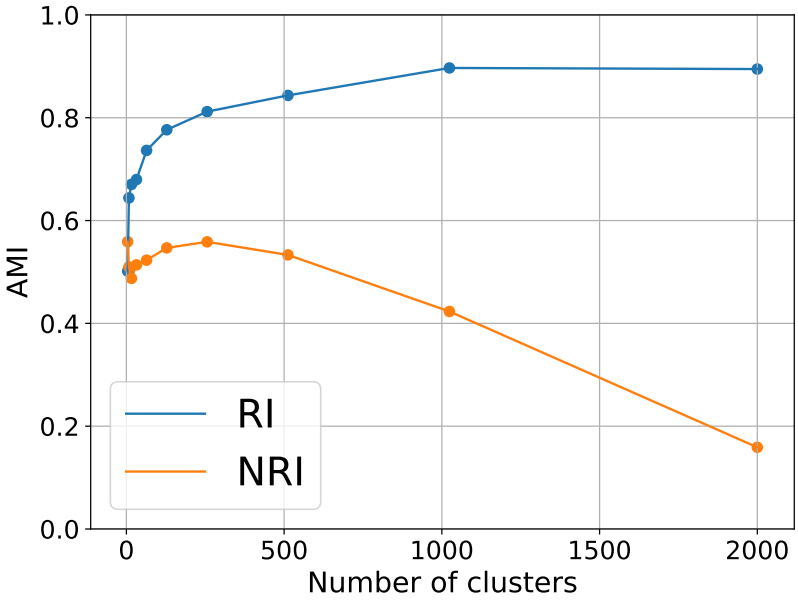


Figure 3.10: Results of multi-cluster test, applied to NRI and RI autoencoders. Clustering agreement scores (AMI) on the `Test` dataset, for from 4 to 2000 clusters. The AMI curve for RICC converges at 0.9, meaning that RICC autoencoder produces rotation-invariant latent representation.

3.8.5 Stability

Figure 3.11a shows that the mean ARI drops from 0.48 at eight clusters to 0.32 at 48 clusters, and then continues to decline to below 0.3 after 68 clusters. Results indicate that there is a presence of cluster assignments consistently seen in different combination of datasets, particularly for stronger agreement observed in $k > 68$.

The significance curve in Figure 3.11b drops to 1.01 at 50 clusters, indicating that our optimal number of clusters is $k^* < 50$.

In Figure 3.11c, the distance metric sharply decreases from 8 to 36 clusters, but the slope then flattens and values are almost unchanged between 40–48 clusters. That is, the pairwise similarity of latent representations drastically increases between 8 and 36 clusters but becomes less different among the range between 40–48 clusters. The selection of a k value from within this range would not change the result significantly. Since test 5.2 provides an upper bound of $k^* < 50$, the results of test 5.3 suggests that the optimal number of cluster

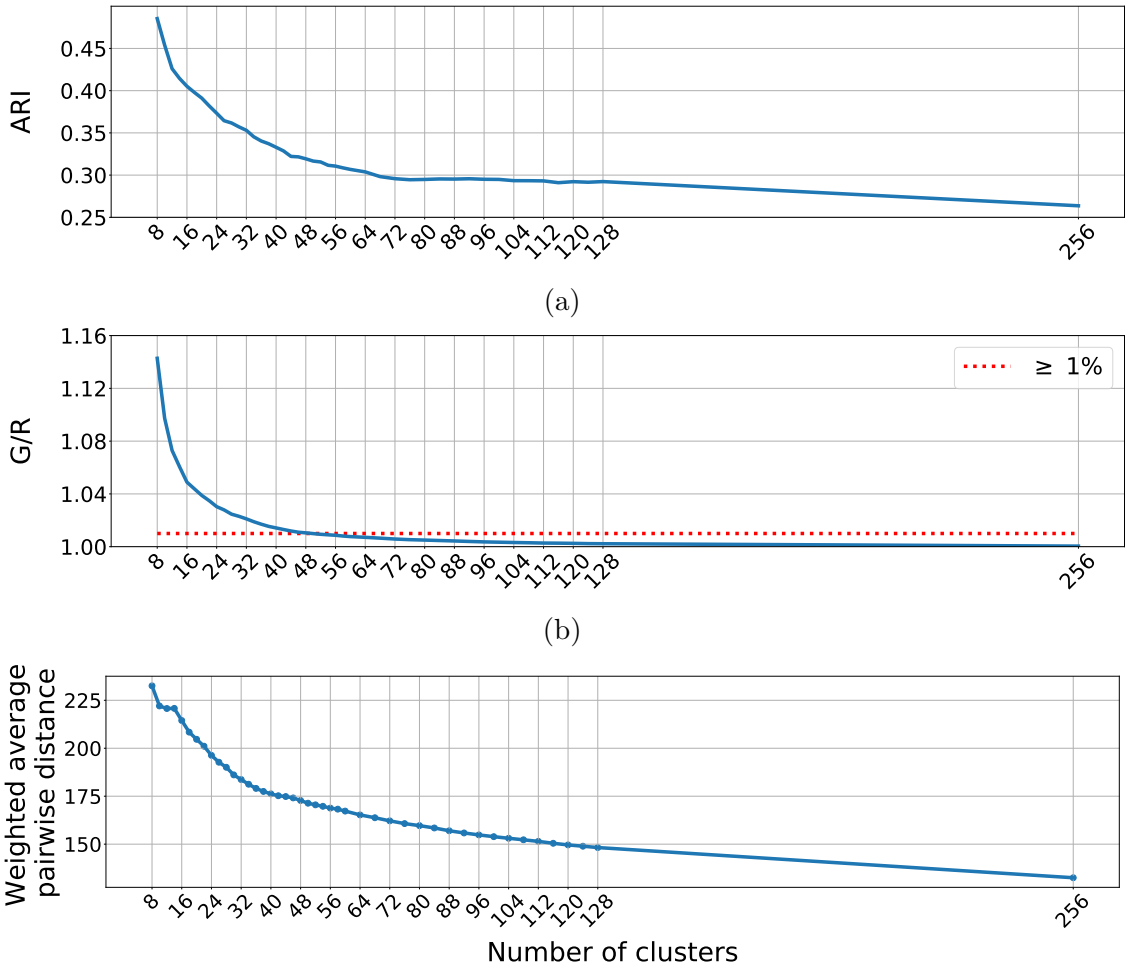


Figure 3.11: Plots for the three stability criteria metrics of Table 3.2, each as a function of number of clusters. (a) *Clustering similarity*: Adjusted Rand Index (ARI) as a measure of similarity of clusterings generated by RICC models trained on different subset of patches. (b) *Clustering similarity significance*: Blue line represents the ratio of the mean Rand Index based on RICC applied to our holdout patches $\{x \mid x \in H\}$ (G) and the mean Rand Index from HAC applied to random uniform distributions (R). The red dashed line is G/R ≥ 1.01, indicating that the stability of cluster label assignments produced from RICC is $\geq$1% better than results of simply clustering random uniform data. (c) *Intra-cluster texture similarity*: Blue line shows the weighted average of the mean squared Euclidean distance between pairs of patches within each cluster. Lower values suggest more homogeneous textures and physical features within each cluster. The use of three similarity tests enables to achieve simultaneously our goal of both stability and maximality when grouping clusters.

lies in $40 \leq k^* \leq 48$.

These are shown as the dots in Figure 3.12. The WASD curve (black) decreases with

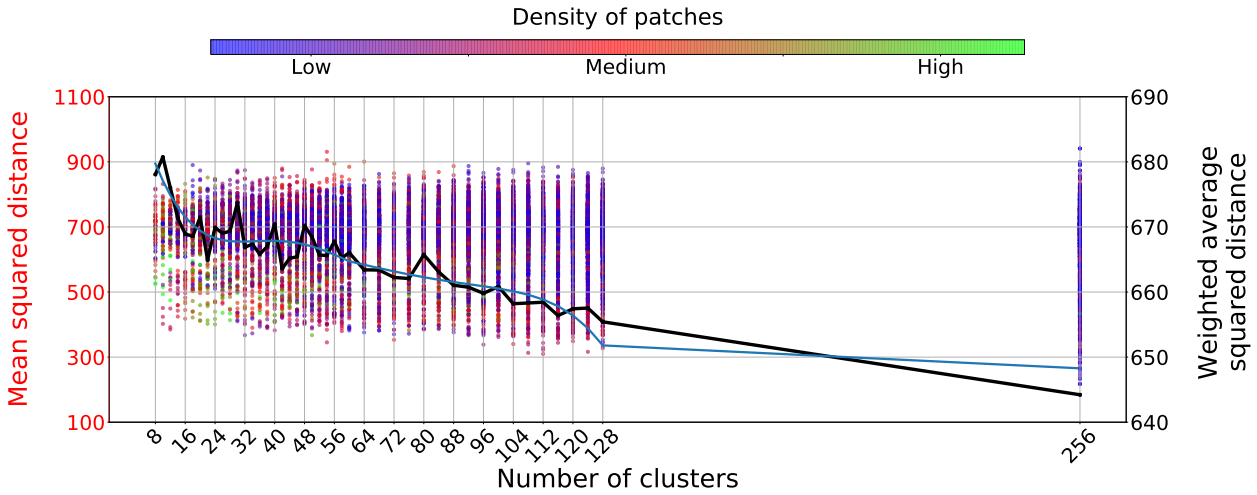


Figure 3.12: Seasonal stability test comparing the intra-seasonal variance of textures within each cluster as a function of number of clusters. Each of $9 \cdot k$ colored dots for each value of k gives the average squared distance (left y-axis) between July and January patches as described in text; the color indicates cluster density, a measure of cluster size. The black line shows the mean WASD (right y-axis) from nine trials as described in text. The blue line shows a smoothed WASD curve obtained by applying a Savitzky-Golay filter with degree six polynomial. The minimum WASD value in $40 \leq k^* \leq 48$ occurs at $k = 42$, motivating our choice for AICCA.

increasing cluster number k, implying as expected that higher cluster numbers allow for better capturing of seasonal changes. Because a smoothed version of the WASD curve (blue) has a minimum of $k = 42$ over the range $40 \leq k \leq 48$, we choose 42 clusters as the optimum number and use this value in the cluster assignment step of Section 4.2. Given that the WMO cloud classes define approximately 28 subcategories, the 42 AICCA clusters should not overwhelm users who use AICCA to investigate cloud transitions.

CHAPTER 4

AICCA: AI-DRIVEN CLOUD CLASSIFICATION ATLAS

The AI-driven Cloud Classification Atlas (AICCA) provides AI-generated cloud class labels for all ocean cloud images that are sampled by MODIS instruments since they started operations and subdivided into 128×128 pixels (∼100 km by 100 km). The cloud labels (ranging from 1 to 42) are generated by RICC and a label assignment scheme (see Section 4.2) designed to generate 42 clusters that are configured via the evaluation protocol (see Section 3.5).

4.1 AI-generated Climate Science Dataset

An AI-generated dataset is a synthetic dataset that replicates characteristics of real data on a large scale to improve the generalizability and performance of AI models. For example, natural language processing traditionally gains greater benefits from synthetic images and/or text to offer more diverse and larger training configurations, leading to a higher model performance [45]. In climate science, generative adversarial neural network (GAN), which is a type of neural network that simultaneously trains two networks to improve the quality of AI-generated synthetic data, produces realistic climate model simulations and images to complement the absence of climate datasets under various scenarios without requiring large HPC resources or observations [121, 138]. Although the synthesized AI-generated data is promising, there is always disputable in the fidelity of the dataset from the "black box" whether it follows the exact first principle of physics. Instead, there are petabytes of underutilized satellite imagery that has been observed over the past several decades, which should be truly addressed by automatic algorithms.

To serve the needs of climate science, AI-generated cloud label datasets can help in the further understanding of cloud feedback and response. CUMULO [189] utilized a combination of spatial satellite cloud images of Aqua instrument from Moderate Resolution Imaging Spectroradiometer (MODIS) and vertical cloud profile and labels from CloudSat to train

a convolutional neural network for producing AI-generated cloud dataset. The approach can classify unlabeled MODIS cloud images into nine cloud types of the International Satellite Cloud Climatology Project (ISCCP). Yet, to our understanding, there is currently no published dataset that offers AI-generated labels and associated metadata for tracking transitions of climate over multiple decades.

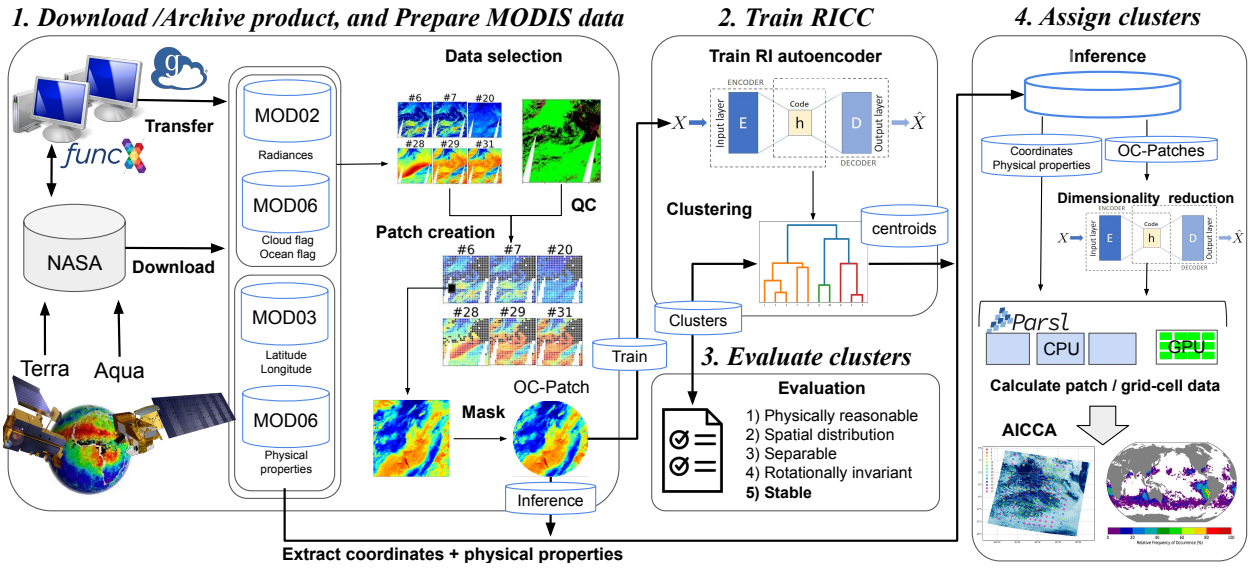


Figure 4.1: The AICCA production workflow comprises four principal stages. **1) Download / Archive and Prepare MODIS data**: Download calibrated and retrieved MODIS products from the NASA Level-1 and Atmosphere Archive and Distribution System (LAADS), using FuncX and Globus for rapid and reliable retrieval of 872 terabytes of three different MODIS products between 2000–2022. Store downloaded data on the Theta filesystem at the Argonne National Laboratory. Select six near-infrared to thermal bands related to clouds and subdivide each swath into non-overlapping 128×128 pixel patches by six bands. Select patches with >30% cloud pixels over ocean regions, and apply a circular mask for optimal training of our rotationally invariant autoencoder, yielding `OC-Patches`. **2) Train RICC**: Train an autoencoder to 1M randomly selected patches to generate latent representations, and cluster those latent representations to determine cluster centroids. **3) Evaluate clusters**: Apply five protocols to evaluate whether the clusters produced are meaningful and useful. **4) Assign clusters**: Use trained autoencoder and centroids to assign cloud labels to unseen data. We use the Parsl parallel Python library to scale the inference process to hundreds of CPU nodes plus a single GPU, and to generate the AICCA dataset in NetCDF format. We then calculate physical properties and other metadata information for each patch and for each $1^\circ \times 1^\circ$ grid cell.

4.2 Assign Cluster Labels

To assign cluster labels for `OC-Patches` unseen in training autoencoder and clustering, we can cluster latent representations from a trained RI autoencoder applied to $\texttt{OC-Patches}_{\texttt{HAC}}$ so as to identify the centroids that will define our cloud clusters for assigning cluster labels. We define the assignment process as follows: Given N data points, a naive HAC approach requires $\mathcal{O}(N^2)$ memory to store the distance matrix used when calculating the linkage metric to construct the tree structure [104]—which would be impractical for the one million or more patches. Thus, we use a smaller set of patches, $\texttt{OC-Patches}_{\texttt{HAC}}$, comprising 74 911 ocean-cloud patches from the year 2003 (the first year in which both Terra and Aqua satellites ran for the entire year concurrently) for the clustering phase. We apply our trained encoder to compute latent representations for each patch in $\texttt{OC-Patches}_{\texttt{HAC}}$ and then run HAC to group those latent representations into k^* clusters, in the process identifying k^* cluster centroids and assigning each patch in $\texttt{OC-Patches}_{\texttt{HAC}}$ a cluster label, $1..k^*$. While we could use a parallelizable HAC algorithm [61, 105, 156] to increase the quantity of data clustered, this would not address the intrinsic limitation of our clustering process given the 872 terabytes of MODIS data.

We have so far trained our RI autoencoder on the 1 million patches in $\texttt{OC-Patches}_{\texttt{AE}}$ and applied HAC to the 74 911 patches in $\texttt{OC-Patches}_{\texttt{HAC}}$ to obtain a set of k^* cluster centroids, $\mu = \{\mu_1, \cdots, \mu_{k^*}\}$, where k^* is the number of clusters defined in Section 3.5. We next want to assign a cluster label to each of the 153 million patches in `OC-Patches`. We do this by identifying for each patch x_i the cluster centroid μ_k with the smallest Euclidean distance to its latent representation, $z(x_i)$. We use Euclidean distance as our metric because our HAC algorithm uses Ward's method with Euclidean distance. That is, we calculate the cluster label assignment $c_{k,i}$ for the i-th patch as:

$$c_{k,i} = \underset{k=\{1,\cdots,k^*\}}{\arg\min} ||z(x_i) - \mu_k||_2. \tag{4.1}$$

This label prediction or *inference* process is easily parallelized. We use the Parsl parallel Python library [7], which enables scalable execution on many processors via simple Python decorators, for this purpose.

4.3 AICCA Patch-Level Data

The AICCA dataset uses all patches from Aqua and Terra MODIS image data during 2000–2022, subject to the constraints that they 1) are disjoint in space and/or time; 2) include no non-ocean pixels, and 3) each includes at least 30% cloud pixels. The resulting set comprises about 153 590 874 individual 128×128 pixel (∼100 km by 100 km) ocean-cloud patches, for each of which AICCA provides the following information (and see Table 4.1):

- `Source` is either Aqua or Terra;
- `Swath`, `Location`, and `Timestamp` locate the patch in time and space;
- `Training` indicates whether the patch was used for training;
- `Label` is an integer in the range 1..42, generated by the RICC configured for 42 clusters based on results from evaluation protocol;
- `COT_patch`, `CTP_patch`, and `CER_patch`, the mean and standard deviation, across all pixels in the patch, for three MOD06 physical values: cloud optical thickness (COT), cloud top pressure (CTP), and cloud effective radius (CER); and
- `CPI_patch`, cloud phase information (CPI), four numbers representing the number of the 128×128 pixels in the patch that are estimated as clear-sky, liquid, ice, or undefined, respectively.

The resulting 146 Bytes per patch represents a 16 159 × reduction in size relative to the raw multispectral imagery. Assignment of cluster labels involves a sorting process that clusters are first sorted on CTP and then on the global occurrence of the clusters within

each 50 hPa pressure bin. Thus, small cluster numbers (e.g., #1) represent high-altitude cloud, and within a similar CTP range (e.g., 500 hPa – 550 hPa), smaller numbers represent the more dominant patterns within the bin. Additional information can assist users in understanding when and where individual patches are extracted from MOD06 by using the patch's geolocation index and timestamp (Location and Timestamp in Table 4.1), and furthermore to locate the patch's data in the appropriate other MODIS product files. These mean values summarize the average physical characteristic for the patch; the standard deviations provide some indication as to the existence of multiple clouds (especially low- and high-altitude clouds). We do not use the MOD06 multilayered cloud flag.

Table 4.1: Information provided in AICCA for each 128×128 pixel ocean-cloud patch: metadata that locate the patch in space and time, and indicate whether the patch was used to train RICC; a cloud class label computed by RICC; and a set of diagnostic quantities obtained by aggregating MODIS data over all pixels in the patch.

Variables	**Description**	**Values**	**Type**
Swath	Identifier for source MODIS swath	1	float32
Location	Geolocation index for the upper left corner of patch	2	float32
Timestamp	Time of observation	1	float32
Training	Whether patch used for training	1	binary
Label	Class label assigned by RICC: integer in range $1..k^*$	1	int32
COT_patch	Mean and standard deviation of pixel values in patch	2	float32
CTP_patch	”	”	”
CER_patch	”	”	”
CPI_patch	Number of pixels in patch in {clear-sky, liquid, ice, undefined}	4	int32

The core AICCA dataset is provided as NetCDF [128] files that combine patches from each MODIS swath into a single file. While AICCA contains no raw satellite data, it includes for each patch an identifier for the source MODIS swath and a geolocation index; thus users can easily link AICCA results with the original MOD02 satellite imagery and other MODIS products. The complete `OC-Patches` set contains around (21 + 23 years) × (365 days/year) × (12 000 patches/day) ×146 B/patch ≈ 26.7 gigabytes, excluding metadata.

4.4 AICCA Daily-Level Data

In addition to providing a set of per-patch data per each swath file, we follow common practice in climate datasets by also providing data composited on a daily basis. The second element of the AICCA dataset spatiotemporally aggregates the patch-level class label and diagnostic values from patch-level data Section 4.3, along with additional informative cloud physical parameters and metadata to provide users with rich information about clouds in a readily applicable format. The selection of additional parameters from MOD03 and MOD06 products are based on the needs for comprehensive analysis of clouds when we conducted interviews with climate scientists.

For each resulting data item, the AICCA daily-level dataset provides the information listed in Table 4.2, a total of 313 Bytes for Aqua and 314 Bytes for Terra:

- `Platform` is either Aqua or Terra;
- `Location` gives a latitude and longitude for the center of the patch;
- `Timestamp` locates the patch in time;
- `Label` is an integer in the range 1..42;
- `COT_patch`, `CTP_patch`, `CER_patch`, `CPI_patch`, are obtained from patch-level data;
- `CWP_patch`, `CE_patch`, `CTT_patch`, `ACWV_patch`, the mean across all pixels in the patch for cloud water path (CWP), cloud emissivity (CE), cloud top temperature (CTT), and the above cloud water vapor (ACWV);
- `CF_patch`, `SIB_patch`, `MLF_patch`,`SF_patch`, the fraction in modulo 100 across all pixels in the patch for cloud fraction (CF), snow and ice background fraction (SIB), cloud multi-layer fraction (MLF), and sunlight fraction (SF); and
- `MLC_patch` is the median of cloud multi-layer confidence (MLC) in the range of 0 to 8 (i.e., higher is more confident) in all pixels in the patch.

Table 4.2: Information provided in AICCA daily-level data: a cloud class label computed by RICC and a set of diagnostic quantities obtained composited for a daily-level file.

Variables	Description	Values	Type
Location	(lat, long) for a patch	2	float64
Timestamp	Time of observation	1	string
Platform	Aqua or Terra	1	string
Label	A class label in a patch	1	int64
COT_patch	Mean and standard deviation of pixel values in patch	2	float64
CTP_patch	”	”	”
CER_patch	”	”	”
CPI_patch	Number of pixels in patch in {clear-sky, liquid, ice, undefined}	4	int64
CWP_patch	Mean of pixel values in patch	1	float64
CE_patch	”	”	”
CTT_patch	”	”	”
ACWW_patch	”	”	”
CF_patch	Percentage of pixel values in patch	”	”
SIB_patch	”	”	”
SF_patch	”	”	”
MLF_patch	”	”	”
MLC_patch	Median of pixel values in patch	”	”

The daily-level AICCA dataset is provided by the comma-separated values (CSV) files that composite pathces from patch-level AICCA dataset into a single CSV file for each day. Because Pandas [113] and Dask [130] that offer built-in support for CSV files are familiar with our end users in climate science community, allowing seamless integration with their analysis workflow. The complete dataset contains around (21 + 23) years $\times$ (365 days/year) $\times$ (12 000 patches/day) $\times$314 B/patch $\approx$ 56 gigabytes.

4.5 Analysis of AICCA dataset

Having configured the optimal number of clusters to generate the AICCA dataset, we now examine whether the AICCA dataset provides useful insight into the process of ocean clouds.

4.5.1 Distribution of clusters

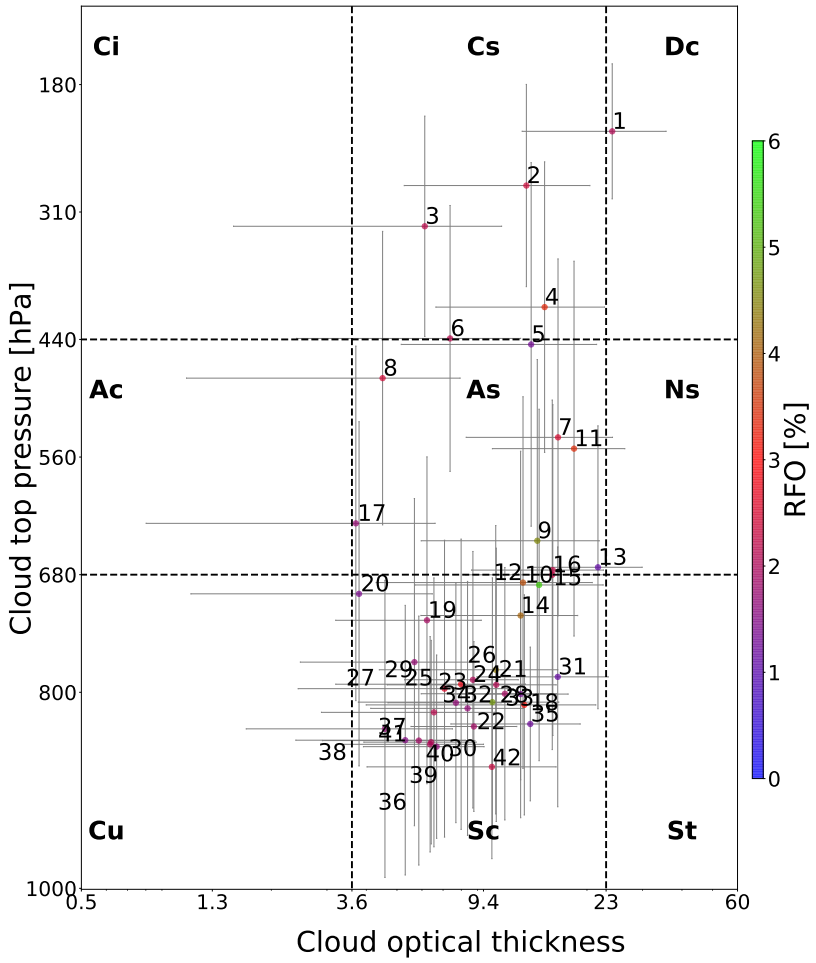


Figure 4.2: Distributions of cluster occurrences and properties from 2000 to 2021 for AICCA in COT–CTP space, where COT is cloud optical thickness (dimensionless) and CTP cloud top pressure (hPa). For comparison, dashed lines divide the nine regions corresponding to the International Satellite Cloud Climatology Project (ISCCP) cloud classes [132, 180]. Dots indicate mean values for each cluster and error bars the standard deviation of cluster properties. Data point colors indicate the relative frequency of occurrence (RFO) of each individual cluster in the dataset. Note that, in assigning cluster labels, we sort the clusters first on CTP and then on the global occurrence of the clusters within each 50 hPa pressure bin. Thus, small cluster numbers (e.g., #1) represent high-altitude cloud, and within a similar CTP range (e.g., 500 hPa–550 hPa), smaller numbers represent the more dominant patterns within the bin.

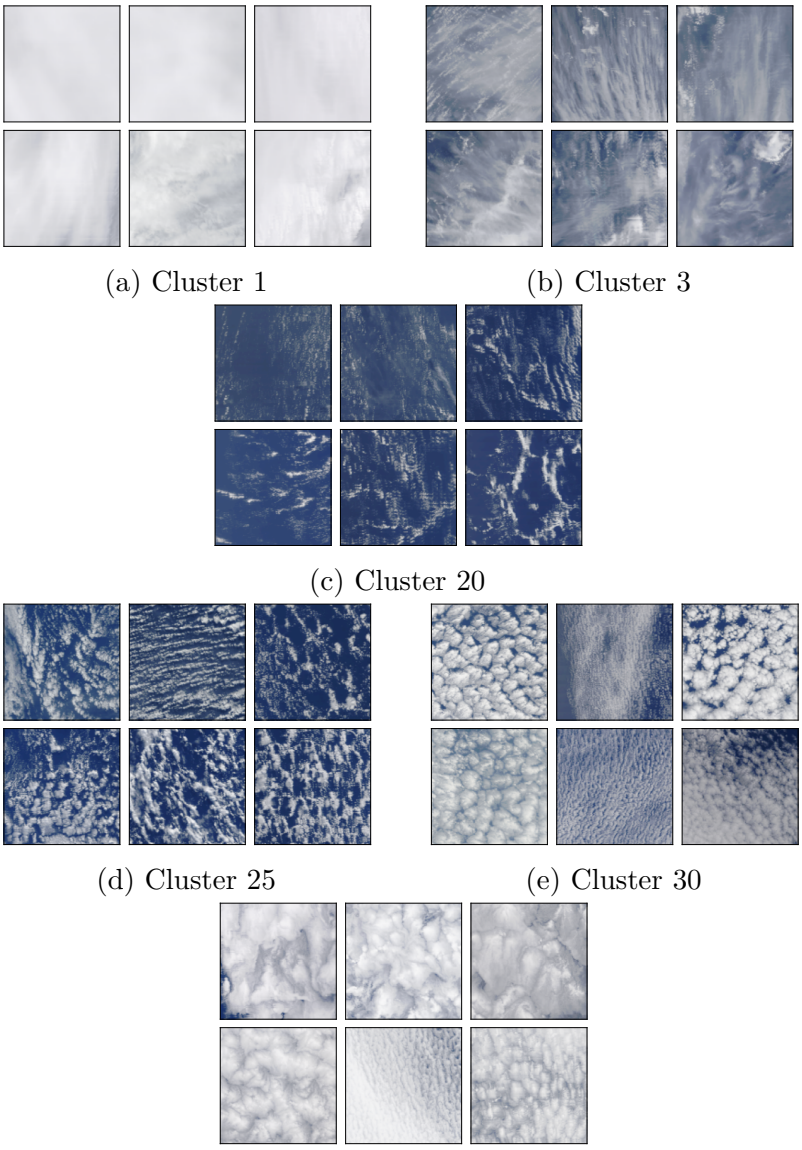
First we investigate the distribution of 42 clusters from AICCA on COT-CTP space. A major limitation of the ISCCP cloud classification scheme is that the variety of cloud textures and physical patterns seen among low clouds are simply merged into a single stratocumulus

cloud, one of the largest concerns for climate scientists. In fact, 49.7 percent of `OC-Patches` from 2000-2021 falls into a single stratocumulus cloud type, where AICCA assigns 30 of 42 classes to the stratocumulus regime. AICCA distinguishes cloud information distinctively at the low cloud altitudes and moderate cloud thickness (Sc), while the classes are distributed on every part of COT-CPT space.

Besides the rich diversity of physical information, we highlight the *texture* distinctions in AICCA cloud classes: Figure 4.3 shows six true color images [44] representing common texture closest to the $\texttt{OC-Patches}_{\texttt{HAC}}$ centroid for each of the six clusters. Patches shown for each cluster are visually similar, and the different clusters have distinct differences. For example, high altitude clouds, #1 and #3 differ in their within-cluster mean optical thickness (#1: 24.1 and #3: 6.1) and differences in their texture correspond the physical features. Similarly, four clusters (#20, #25, #30, and #35) that fall into the traditional stratocumulus cloud category in ISCCP match their mean optical thickness and their sparse/dense cloud textures. These distinctions show that AICCA is separating stratocumulus clouds by texture as well as by mean properties across the patch.

4.5.2 Geographic Distribution of Cluster Label Occurrence

In this study, we leverage AICCA daily-level dataset (see Section 4.4) to examine the geographic distribution of AICCA cluster labels from 2000 to 2022. Figure 4.4 shows mean incidences for each of the 42 cloud types in the dataset, gridded on a 1° global grid to calculate relative frequency of occurrences of each of 42 clusters. Results exhibit strong geographic distinctions among cluster labels, with some occurring only in the tropics and others only at high latitudes. Some show even finer geographic restrictions. For example, cloud classes #1–#3 are localized primarily in the West Pacific warm pool, all likely associated with tropical deep convection, though ranging in altitude (227.7–324.6 hPa CTP) and thickness (24.1–6.1 COT). Note again that classes are numbered in order of their mean altitude; see Section 4.2 for details. By contrast, the stratocumulus cloud labels discussed for Figure 4.2 show different



(a) Cluster 1 (b) Cluster 3

(c) Cluster 20

(d) Cluster 25 (e) Cluster 30

(f) Cluster 35

Figure 4.3: Selected MODIS true color images [44] for the six clusters that dominate high altitude clouds (#1 and #3), sparse and open-cell stratocumulus (#20 and #25), and closed-cell stratocumulus (#30 and #35) clouds. Surtitles show the cluster numbers. We show the six representative patches closer to $\texttt{OC-Patches}_{\texttt{HAC}}$ centroids. The example patches indicate that AICCA discriminates well between textures (e.g., compare the fine-scale detail of #20 to the more coarsely aggregated #35) even for patches of similar mean cloud properties seen in Figure 4.2.

distributions. Those most clearly associated with classic closed-cell stratocumulus—#30 and #35—are as expected primarily localized to small areas on the west coasts of continents. The most predominant open-cell stratocumulus cloud, #25, is more widely distributed but with

strong latitudinal dependence. The three clusters described are all low in altitude (mean CTP of 796.3–834.9 hPa) and moderate in thickness (mean COT of 8.8–13.2 thickness for the closed-cell classes and 7.1 for the open-cell). Therefore all would be labeled as Sc in the ISCCP classification, whereas AICCA reveals their striking differences. Additionally, it is worth noting that when comparing the geographical distributions with different seasons, the geographic distinctions become even sharper patterns along with migrating seasonally with the sun's position.

The strong localization of some cloud classes near the poles raises concern that they may be affected by the presence of sea ice. We have restricted analysis to ocean clouds to avoid the complications of surface effects—the ocean provides a dark and homogeneous background—but parts of the high-latitudes ocean are covered in wintertime ice. Because two of the MODIS bands used in our cloud clustering system, bands 6 (1.6 μm) and 7 (2.12 μm), are also used by the MODIS snow and ice detection algorithm [129], the resulting AICCA dataset can inadvertently include some surface background information in the latent representation. To check for contamination, we use a MODIS cloud product that describes the presence of a snow and ice background for each pixel (MOD06). Only one cloud class may experience significant interference: #12, which forms in local winter. (Sea ice makes up 16/31% of its labeled pixels in January/July.) The other polar cloud classes appear in local summer. Sea ice effects therefore do not appear to drive the labeling of geographically distinct cloud classes that appear in polar oceans.

These results suggest that AICCA identifies real and important differences between cloud types and can help climate scientists understand the drivers of distinct cloud patterns and regimes.

4.5.3 Trends in subtropical stratocumulus

For our case study, we consider whether the AICCA classes can provide insight into a recently discovered cloud trend: low cloud cover has been decreasing in the Pacific Ocean

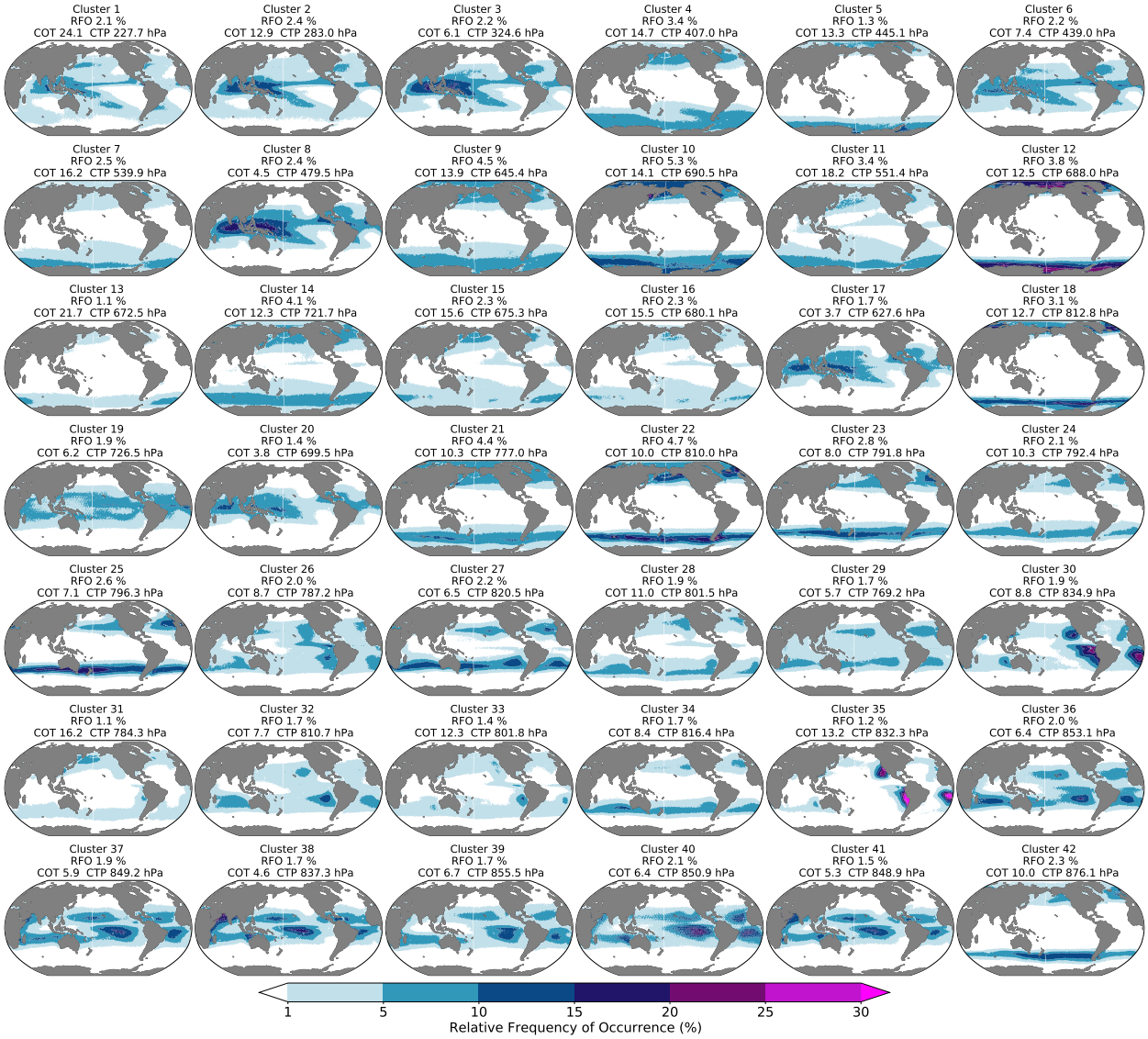


Figure 4.4: An example application of AICCA. We plot the relative frequency of occurrence (RFO) for each of the 42 AICCA_{42} clusters, using all data from 2000 to 2021. Land is in grey, and areas where RFO $< 1.0\%$ are in white. Surtitles show global mean RFO, cloud optical thickness (COT), and cloud top pressure (CTP) for the given cluster. Clusters show striking geographic distinctions, and those with roughly similar spatial patterns have different mean physical properties, suggesting meaningful physical distinctions.

off the coast of Southern California and the Baja peninsula [1]. This finding was based on mean MODIS cloud properties alone, with no consideration of cloud patterns.

Figure 4.5 shows the trends in 18 years (2003–2021) of MODIS observations off the S. California coast for eight cloud classes. We exclude 2000–2002 because the Aqua and Terra instruments were not operating simultaneously over an entire year during that period.

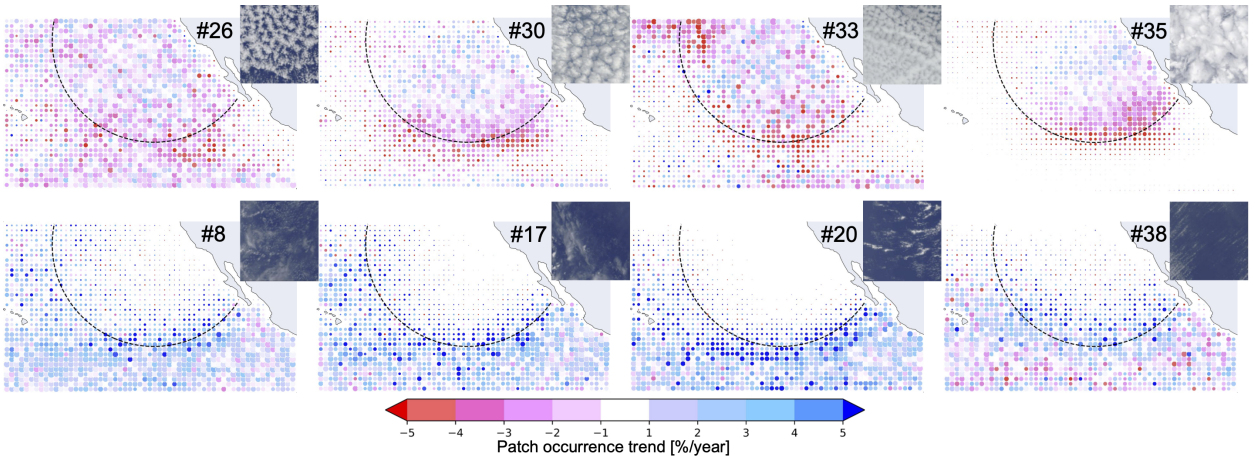


Figure 4.5: Trends in occurrence of selected AICCA cloud classes over the subtropical N. Pacific Ocean (100–160W and 5–40N). Colors show fitted linear trends over the 18 years 2003–2021, expressed in units of % of the mean value over this time. Dot size represents mean relative frequency of occurrence within a class. Dashed curve highlights the approximate edge of the deck.

Closed-cell (or transitional) stratocumulus are the most dominant cloud types in this part of the Pacific, making up a quarter of all cloud occurrences (top row, #26, 30, 33, 35, which are four of the five most predominant classes). All these classes decrease strongly, especially along the southern edge of the deck—by as much as 2/3 in some locations. However, several classes of optically thinner clouds increase, especially just south of the deck region (bottom row, #8, 17, 20, and 38). These classes represent very sparse cumulus or alto-cumulus, similar in visual texture but slightly higher in altitude. The combined effect is that sparse classes infiltrate the stratocumulus deck along its unstable edges. (The same analysis applied to ISCCP cloud classifications also shows a decrease in stratocumulus and increase in cumulus clouds, but AICCA cloud classes provide finer details on these transitions.) This trend may be temperature-driven: stratocumulus decks form only when atmospheric conditions are highly stable, and warmer sea surface temperatures tend to reduce stability. Sea-surface temperatures in the region from the ERA5 reanalysis [49] do increase by nearly 1K over this time period.

The AICCA dataset also lets us examine the temporal evolution of cloud patterns over shorter timescales. For now we consider evolution at fixed (Eulerian) locations, ignoring

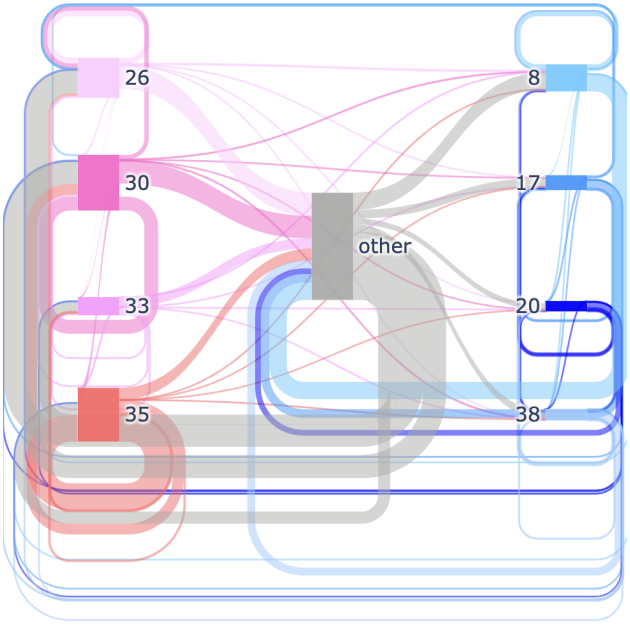


Figure 4.6: Sankey diagram of sub-daily and daily cloud class transitions within gridcells for the 2003–2022 period for the eight classes, and region, in Figure 4.5. Box widths represent occurrence frequency of each class, and colors are ordered by trend. Ribbon width represents the rate of transitions from one class type to another. Loops represent persistence of a class from one time period to the next.

advection by winds. (Stratocumulus textures are driven by a mix of local environmental control and horizontal advection, but the Eulerian perspective can provide insight at sub-daily to daily timescales.) Figure 4.6 shows, as a Sankey diagram, transitions from one time period to the next ($\sim$ daily) for the selected classes and region shown in Figure 4.5. We see significant evolution within closed-cell stratocumulus. The most frequently occurring class, #35, is unsurprisingly also the most stable, and evolves primarily into other closed-cell forms (#30 and #26), which in turn transition into a wide variety of classes. Less than 10% of transitions represent direct evolution from selected closed-cell to selected sparse cloud classes (though still more than the reverse direction). That is, transitions from closed cell classes to sparse cloud classes are largely modulated through a complicated network of mixed type classes. This result suggests that the replacement of closed-cell by sparse cloud types in Figure 4.5 does not simply result from a reduction in stratocumulus lifetime.

CHAPTER 5

SCUBA: SELF-SUPERVISED CLOUD BIAS ASSESSMENT

In this thesis, I describe the Self-supervised Cloud Bias Assessment (SCuBA) framework that validates biases included in simulated clouds from high-resolution numerical climate simulations. Specifically, the framework compares representations of simulated clouds against those from satellite observation in terms of the differences in physical properties, structure, and textures.

5.1 Related Work

Advances in computing power for HPC systems greatly encourage scaling high-resolution simulations with 3–5 km horizontal grid resolution, enabling to compute more accurate simulated large-scale cloud phenomena (e.g., storm and Madden-Julian Oscillation) [152]. However, even the state-of-the-art 3–5 km simulations [158] are not fully resolving deep convection, leading to biases in simulated clouds. Given that the clouds play one of the largest uncertainties in climate projections [191], a universal bias assessment application that can assess the fidelity of simulated clouds is necessary. The SCuBa therefore aims to serve as a validation tool for the purpose of a better understanding of clouds and fine-scale processes generated from microphysics and radiation schemes in high-resolution climate models, particularly comparing against satellite observations.

Self-supervised learning is now introduced for comparing biases in climate models. Mooers et al. [106] use a variational autoencoder and clustering approach to systematically group biases and different convective features among several global storm-resolving models. The self-supervised method can provide less susceptible to human biases when evaluating models as well as novel insights as a combination of physical schemes and model configurations. The SCuBA framework also has potential advantages over existing supervised learning-based model bias scheme [112] without reliance on limited location, time/season, and selection of

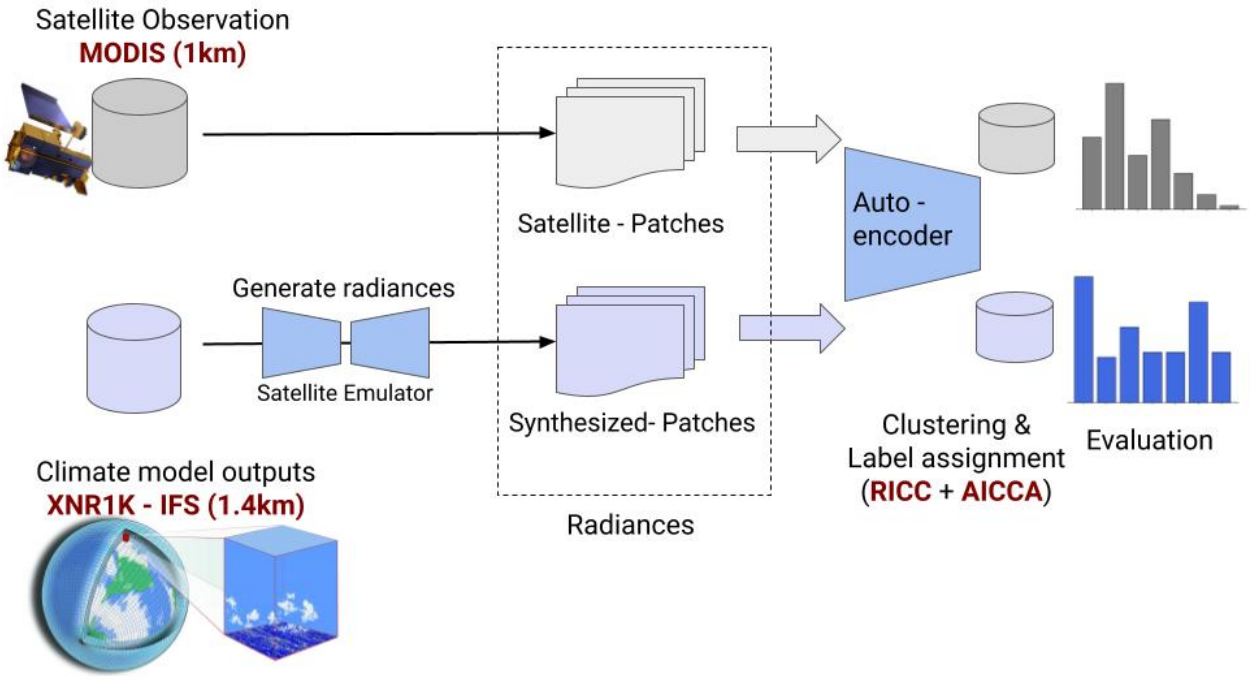


Figure 5.1: Diagram of the workflow of SCuBA.

climate models by employing a trained RI autoencoder to produce latent representations and using the class assignment scheme described in Section 4.2 to assign cluster labels for patches generated from climate models.

5.2 Methodology

In this section, I describe the procedure of SCuBA framework and developments of MODIS satellite emulator.

5.2.1 Workflow of SCuBA

Figure 5.1 illustrates the overall workflow of SCuBA framework. For variables from outputs of climate model, let ϕ be selected variables to be used for training and testing satellite emulator M. Here, we construct a deep learning emulator that emulates transfer radiance calculations without expert knowledge such as the Satellite Data Simulator Unit (SDSU) [93]. The more detail is discussed in Section 5.2.2. Having pre-processed model

Algorithm 3 Pseudocode for the self-supervised cloud bias assessment.

Input: ϕ: { variables of climate model used for emulator}
Output: $\tilde{C} = \{\tilde{c_1}, \cdots, \tilde{c_{42}}\}$: AICCA class labels via synthesized radiances.

1: $\tilde{X} = M(\bar{\phi})$ where M : satellite emulator ▷ Synthesize MODIS radiances via emulator
2: $z(\cdot)$: { layers in autoencoder to map input x to latent representation}
3: **for** $\tilde{x_i}$ from $\tilde{X}$ **do**
4: $\tilde{c}_{k,i} = \underset{k=\{1,\cdots,42\}}{\arg\min} ||z(\tilde{x}_i) - \mu_k||_2$ where μ: centroids ▷ Assign cluster label for the i-th patch based on synthesized radiances
5: **end for**
6: Return cluster distribution of $\tilde{C}$ ▷ Assign one of 42 labels

variables listed in 5.1, we train and execute satellite emulators to generate synthesized MODIS radiances for bands 6, 7, 20, 28, 29, and 31 and subsequently patches, giving $\tilde{X}$. We then assign one of AICCA class labels (line 4) via a trained RI autoencoder, giving a distribution of class labels $\tilde{C}$ to compare ones from AICCA dataset from real MODIS observation.

To validate the biases in simulated clouds from climate models, we compare the distributions of AICCA class labels generated from real MODIS radiances and synthesized ones. Through this comparison, SCuBA can gain insight into the sources of bias and further refine the microphysics process in tested climate models.

5.2.2 Satellite Emulator

The application of a satellite simulator allows for the generation of synthesized radiances and thus patches as input to the RI autoencoder, thereby independent of the underlying climate models. However, satellite simulators require substantial expert knowledge for their tuning to generate realistic radiances. Differences in microphysical schemes among different models also influence the fidelity of output radiances from simulators. Therefore, along with the development of SCuBA framework, we develop a deep learning-based satellite emulator to learn the mapping between climate variables and radiances without appropriate assumptions of microphysics. Here, we define the emulator as a deep learning algorithm, that maps a set

of variables from numerical simulations to radiance through learning the radiative transfer function without scattering and microphysics hypothesis. Ultimately, the development of the emulator can lower the bar to simulators for non-experts.

Table 5.1: MODIS products and IFS variables used to create a training and testing dataset. As noted in the text, a product name *MOD02* in the first column refers to both the Aqua (MYD0X) and Terra (MOD0X) products. Convective and stratiform rain index is not a default variable within IFS and thus we calculate the index based on convective and stratiform rain rate from IFS. Source: NASA Earthdata; ECMWF.

Product	Description	Band	Primary Use
MOD02	Shortwave infrared (1.230–1.250 μm)	5	Land/cloud/aerosol properties
	Shortwave infrared (1.628–1.652 μm)	6	Land/cloud/aerosol properties
	Shortwave infrared (2.105–2.155 μm)	7	Land/cloud/aerosol properties
	Longwave thermal infrared (3.660–3.840 μm)	20	Surface/cloud temperature
	Longwave thermal infrared (7.175–7.475 μm)	28	Cirrus clouds water vapor
	Longwave thermal infrared (8.400–8.700 μm)	29	Cloud properties
	Longwave thermal infrared (10.780–11.280 μm)	31	Surface/cloud temperature
IFS	Land-sea mask		Proportion of land/ocean in a grid
	10 meter U		Eastward component of 10m wind
	10 meter V		Northward component of 10m wind
	Surface pressure		Natural logarithm of surface pressure
	Skin temperature		Temperature of the surface of the Earth
	2 meter temperature		Air temperature at 2m above the surface
	Total column cloud liquid water		Liquid water in cloud droplets in a column
	Total column cloud ice water		Ice contained within clouds in a column
	Total column rain water		Water within rain droplets in a column
	Total column snow water		Snow in a column
	Total column vertically-integrated water vapour		Water vapour in a column
	Convective/stratiform rain index		Convective or stratiform rain

XNR1K is a simulation run at the global 1 km resolution based on the Integrated Forecasting System. The simulation targets the northern hemispheric winter months (NDJF) initialized at 00:00 UTC on 1 November 2018 and the other for the North Atlantic tropical

cyclone season (ASO)initialized at 00:00 UTC on 1 August 2019 with output every 3 hours. We select 11 variables from XNR1K outputs based on inputs required to the Satellite Data Simulator Unit (SDSU) [93]. We select 2-dimensional variables at surface and column values shown in Table 5.1. To compute the convective and stratiform index, we determine three indices; we assign no cloud/rain as 0, convective rain as 0.5, and stratiform rain as 1.0 based on convective cloud rain rate and stratiform cloud rain rate from IFS outputs. As radiance values are influenced by surface conditions, we add land-sea mask. It also helps to extract ocean-only data for downstream task to match patches used for RICC and AICCA. Following the standard practice, we normalize both MOD02 and IFS data in the range 0 to 1. For training a satellite emulator, we select the first 16 hours after model initialization. This data selection could involve pixel values that reveal the displacement of cloud types and the unrealistic amount of rain or snow due to the non-equilibrium states during initialization. Whereas these initial parts of numerical simulation typically spin up to reach equilibrium states and are not used in post-processing, we use them to match cloud structures in IFS and real MODIS observations. To align with MODIS patches, the IFS dataset is also subdivided into 128 pixels $\times$ 128 pixels smaller scale of images. We generate 75 000 training and 5000 testing images, which contain a pair of 128 $\times$ 128 pixels $\times$ 6 channels of MOD02 radiances and 128 $\times$ 128 pixels $\times$ 12 variables of IFS data.

ML-based radiative transfer models are either conventional ML approaches (e.g., random forest [60]) or 1D-CNN [111], which learn a mapping function between a set of climate variables and radiances at each pixel. Since our data might not always contain physically correct rain and snow values, the pixel-based approach could be susceptible to these inconsistencies of physics. Rather, we leverage 3D-CNN to approximate radiances over a certain spatial area. Figure 5.2 shows the U-Net model architecture used for our experiment. After we increase the filter size from 12 to 64 via the first convolutional layer, we halve the horizontal image size every two layers of convolutions but double the filter size from 64 to 512 at the bottleneck. We train U-Nets for 400 epochs using the decoupled weight decay

Table 5.2: Mean Square Error(MSE)

Band	6	7	20	28	29	31
MSE	0.0233	0.0307	0.00872	0.0016	0.0034	0.0030

regularization Adam optimizer [86] with learning rate 3×10^{-4} to predict individual MODIS channels. We select batch size 32 to fit the data into a NVIDIA V100 GPU.

As a quantitative test of trained emulators, we examine mean square errors between MODIS and synthesized radiances for 5000 test images. Results in Table 5.2 reveal that emulators can reconstruct radiances at thermal bands (band 20, 28, 29, and 31) at MSE $\mathcal{O}(1 \times 10^{-3})$ and near-IR bands also achieve to produce radiances with sufficiently small errors (0.0233 for band 6 and 0.031 for band 7) emulators can produce radiances with sufficiently low errors. Whereas bands 6 and 7 have relatively higher errors than those in other bands, Figure 5.3 qualitatively suggests the decent fidelity of reconstructions even while the reconstructed radiances are blurred and lose the high-frequency information. These results indicate that the trained emulator modestly approximates radiances from a set of climate variables.

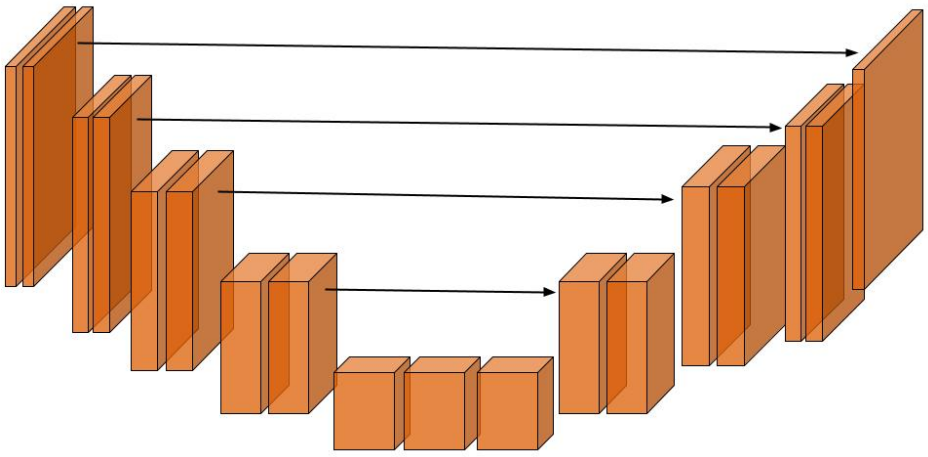


Figure 5.2: Architecture of UNet. An orange box represents a convolutional filter and a black arrow represents a skip connection. We halve the horizontal image size every two layers of convolutions but double the filter size from 64 to 512 at the bottleneck.

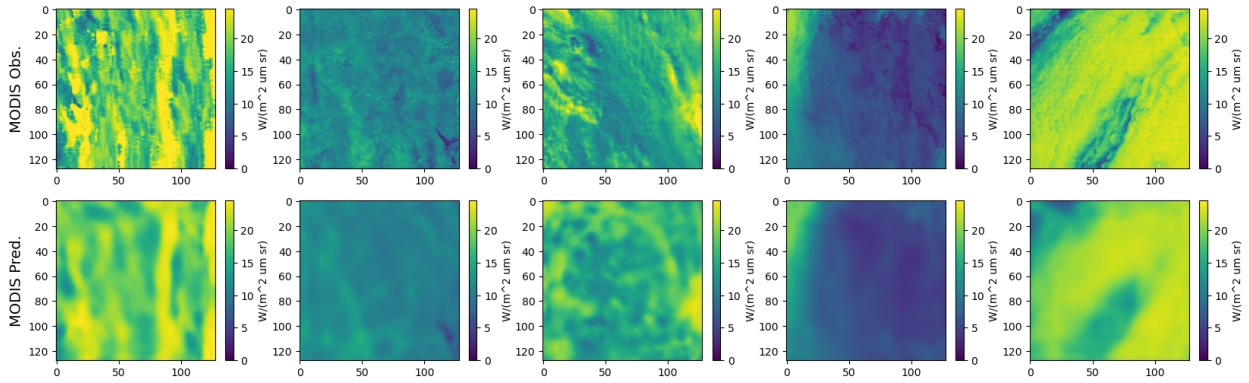


Figure 5.3: Four pairs of example patches of real MODIS band 6 (top row) and generated band 6 (bottom row).

5.3 Model cloud representations

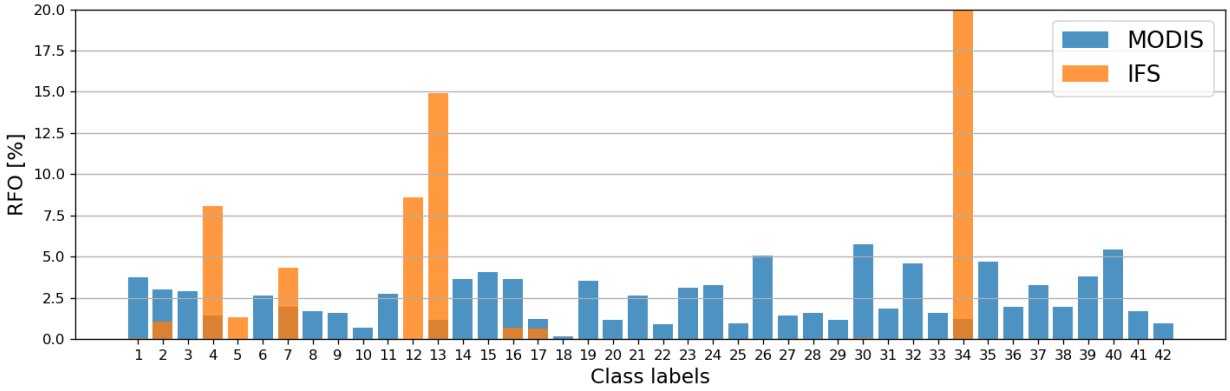


Figure 5.4: Bar plots to compare the frequency of occurrences of 42 clusters from AICCA dataset and cloud classes resulting from synthesized radiances based on IFS.

I first examine the relative frequency of occurrences (RFO) of 42 cloud class assignments from synthesized MODIS radiances and those real observations from the AICCA daily dataset. Figure 5.4 exhibits that 42 cloud classes produced from MODIS satellite images have a variety of classes that vary the RFO between approximately 1 to 5 % in their major classes, except class #5 (no patch was assigned) and #18 (RFO is less than 1%) in the target region. It is noteworthy that low cloud classes with distinct texture features such as # 30, #32, and #35 show 5.3%, 4.8%, and 4.9% of RFO respectively, accounting for more percentage of RFOs than other low-cloud classes. This indicates the diversity of cloud texture

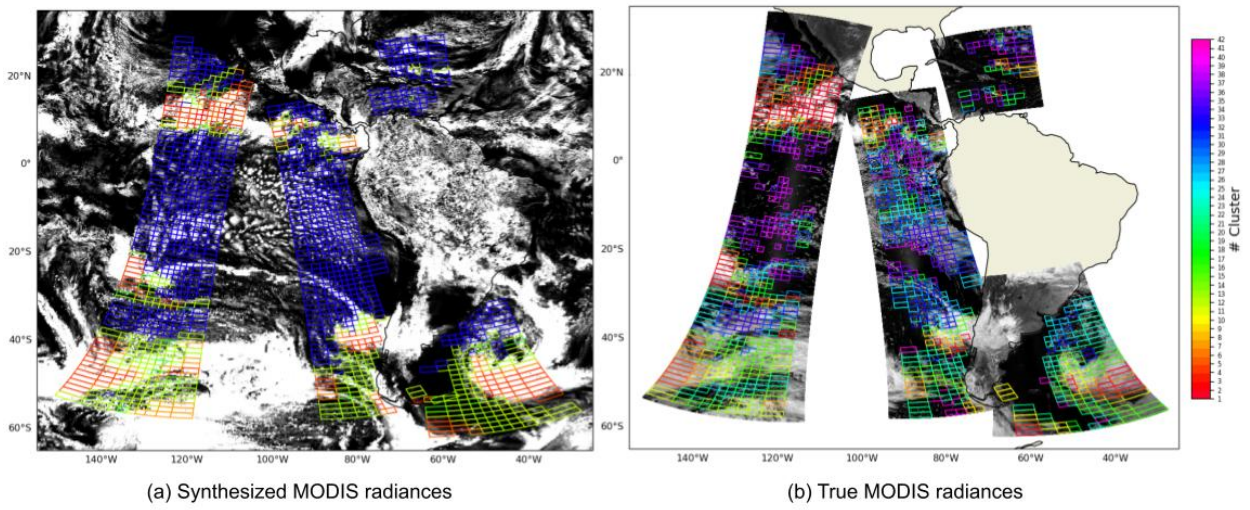


Figure 5.5: Snapshot of 42 AICCA cloud classes assigned to patches from synthesized and real MODIS radiances. (a) spatial class distributions based on synthesized MODIS radiances from trained emulators. The background image is the total column liquid water and ice water from 2018-11-01 18:00 UTC. (b) spatial class distributions based on true MODIS radiances. The background image is band 2 (0841 - 0.876 µm) collected from Aqua and Terra swaths 2018-11-01 17:40 – 19:35 UTC. Color depicts 42 cloud classes: warmer colors indicate smaller class numbers or high-altitude clouds, and colder colors indicate larger class numbers or low-altitude clouds. Cloud classes from synthesized radiances lack their diversity of cloud classes, indicating that cloud representations in climate models are different from real observations.

and structure in this region from MODIS images. In contrast, 42 cloud class assignments based on synthesized radiances generated from the emulator with IFS show only 9 classes at the same target area. In particular, I observe that all low clouds in synthesized radiances are categorized in class # 34, which accounts for 65% of RFOs and also is not often seen in the observation. Other high- and medium-altitude cloud classes (i.e., # 4, # 12, and #13) also take the major percentage of those clouds. To compare their spatial distribution, I map class assignments and their cloud structure as a background image in Figure 5.5. Note that I use cloud water and ice path from IFS as the background cloud image for results from IFS in Figure 5.5(a) and MODIS band 6 (near-IR) for results from MODIS in Figure 5.5(b). The location of low/medium/high clouds between IFS and MODIS roughly matches but as Figure 5.4 represents, the model low-clouds lack their diversity for the class assignments

These results raise a question: why do cloud classes from synthesized radiances based on IFS show less variety of 42 class distributions? The question may stem from two potential factors: (a) the cloud representation within the model may be oversimplified or lack diversity, and/or (b) limitations in the satellite emulator may be linked to the blurred reconstructions of radiance data. I next investigate these two factors in the following paragraphs.

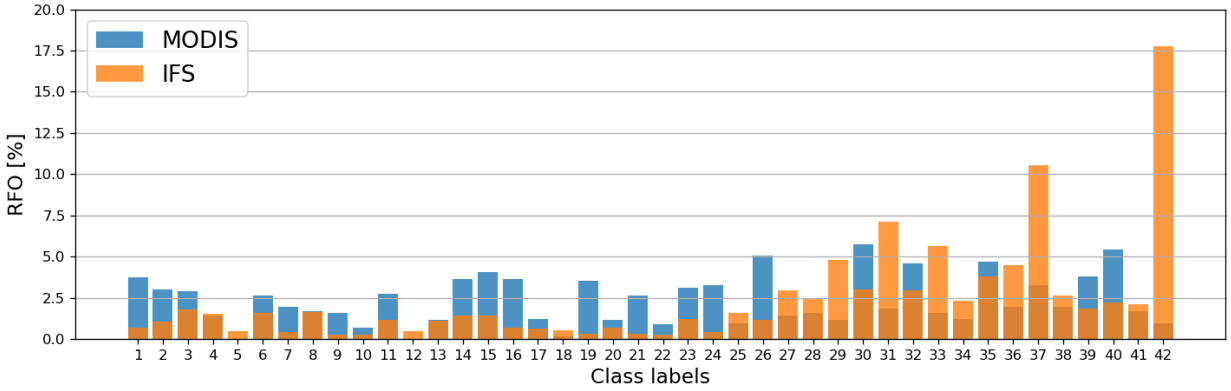


Figure 5.6: Similar to Figure 5.3 but results computing from synthesized radiances based on a UNet model trained with synthesized radiances themself.

To address this question, I conduct an additional test wherein I reassess the occurrences and spatial distributions of cloud classes from RICC, which is trained solely based on the synthesized radiances comprising 75 000 patches. Following the training process, I apply HAC to these patches to create 42 cloud classes. It is important to note that these 42 cloud classes no longer correspond to those from the AICCA dataset. This test facilitates evaluations to determine whether IFS accurately represents the diversity of cloud classes within its model dynamics or if it exhibits limitations in cloud representation. Subsequently, I investigate the class distributions within the same test region over the East Pacific.

Figure 5.6 and Figure 5.7 illustrate the relative frequency of occurrences and the spatial class distributions over the East Pacific, respectively. The results of our analysis reveal significant differences in cloud class generation when using synthesized radiances from the IFS model compared to true MODIS cloud patches. Specifically, the 42 cloud classes generated from RICC trained with synthesized IFS radiances exhibit a wide variety of classes, covering

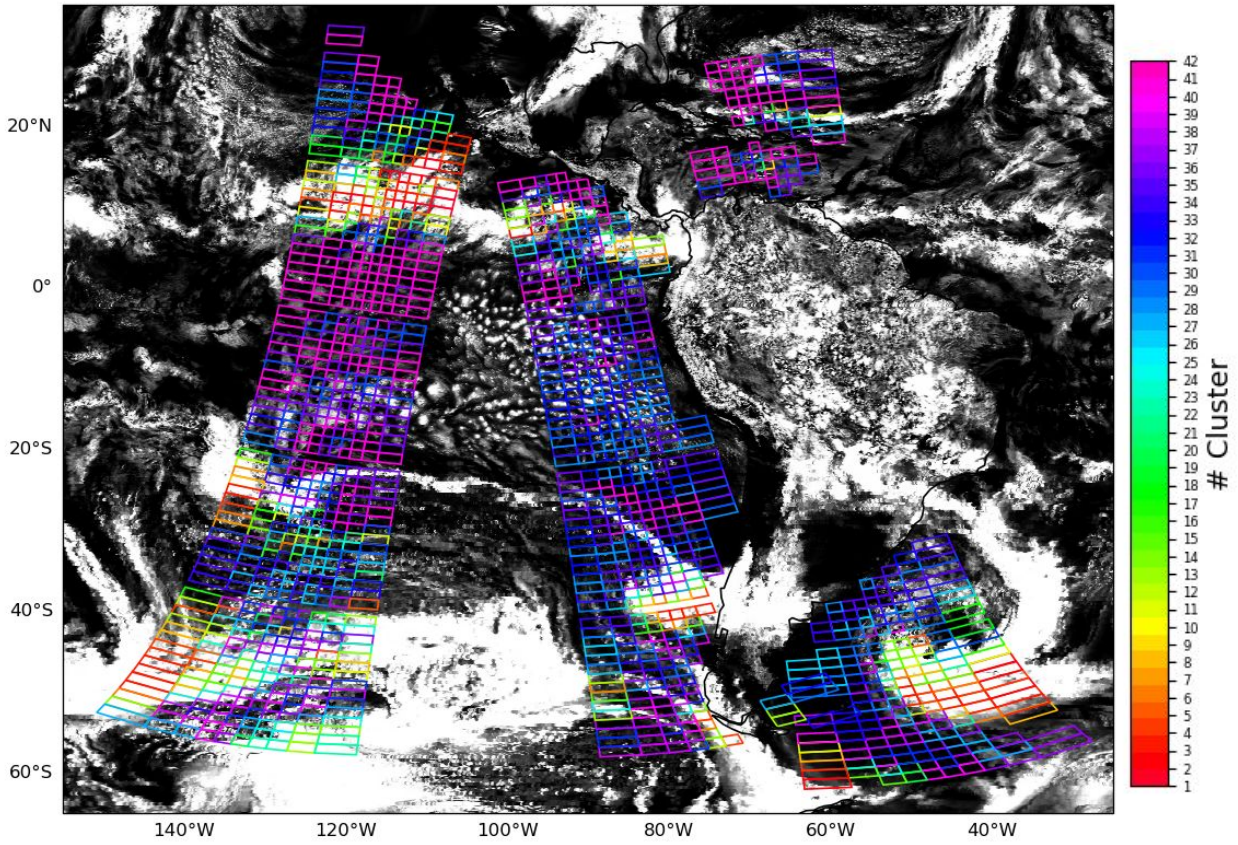


Figure 5.7: A snapshot of spatial distributions of cloud classes generated from RICC trained on synthesized radiances. The map shows locations of 42 cloud classes colored in orange in Figure 5.6.

nearly all 42 classes, indicating a diverse representation of cloud types. In contrast, feeding the synthesized radiances data to RICC trained with true MODIS cloud patches yields results that are quite distinct or opposite. Moreover, when comparing the RFOs, the IFS model generates a higher proportion of low cloud classes compared to MODIS, with clusters #31, #33, and #37 comprising 7.3%, 5.5%, and 11% of RFOs, respectively. Notably, class #42 produces 17.7% of RFOs, indicating a significant presence of very low clouds from the IFS model. Spatial distribution analysis further validates the diversity of class assignments, with stratocumulus cloud decks off the coast of Chile divided into several low cloud classes and high clouds over the Intertropical Convergence Zone exhibiting various high to medium

altitude cloud classes at the edge. These findings suggest that RICC, trained exclusively on synthesized radiances, captures distinct clusters, reflecting the inherent diversities in the high-resolution model.

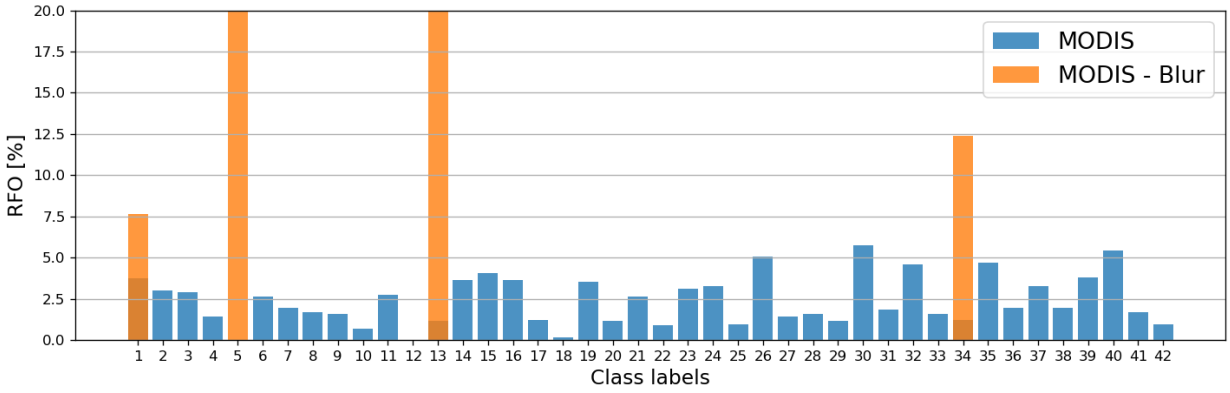
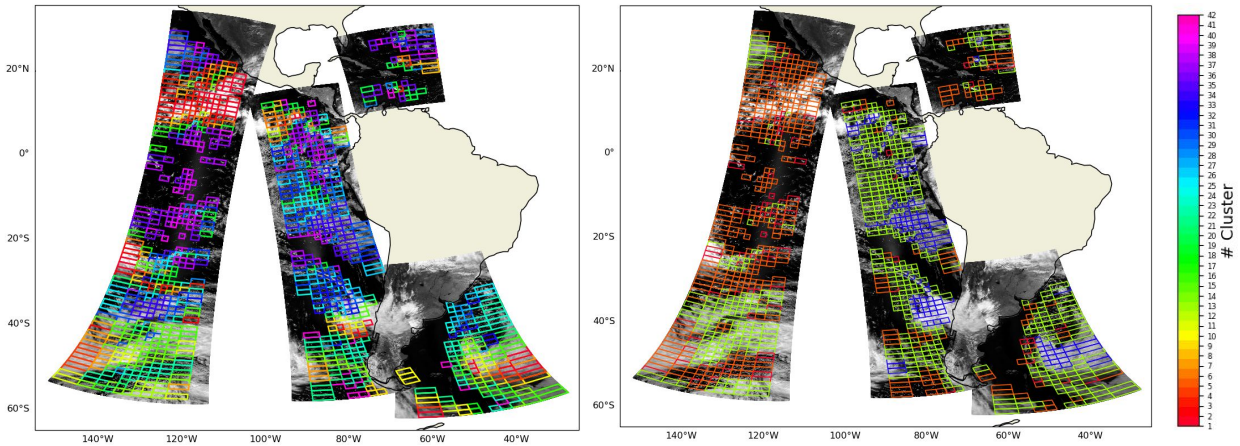


Figure 5.8: Bar plots to compare the frequency of occurrences of 42 clusters from AICCA dataset and cloud classes resulting from RICC algorithm based on blurred MODIS radiances.



(a) MODIS Original (b) MODIS Blurred

Figure 5.9: Similar to Figure 5.5 but comparison of spatial distribution of 42 cloud classes resulting from AICCA dataset (right) and RICC that is trained with blurred MODIS cloud patches. The blurring radiance within patches from MOD02 significantly alter the spatial locations and patterns of assignment of 42 cloud classes. The results indicate that the RICC algorithm captures input spatial patterns in detail so that the radiance emulator needs to generate high-resolution predictions.

To investigate the limitation of the satellite emulator, I newly train the RICC algorithm with 75 000 cloud patches that are blurred by gaussian filter. When assigning class labels, I

use the centroids used for creating AICCA dataset. As the CNN-based emulator truncates high-frequency modes that lose fine texture structure in original MODIS cloud images, the test serves to quantify to what extent the capability of the satellite emulator may limit the performance of this framework. The blurring operation uses gaussian filter by setting a standard deviation as 3, which produces quantitatively similar blurred images reconstructed by the emulator.

Figure 5.8 shows the frequency of occurrences of clusters derived from 42 classes obtained from MODIS and blurred MODIS images. The blurring process introduces significant discrepancies in resulting class assignments, even for original MODIS cloud patches (i.e., patches without blurring). The results reveal a limited range of resulting class assignments, indicating that those observed from synthesized radiances from IFS, with only four classes (#1, #5, #13, and #34) prominently represented. Figure 5.9 illustrates the comparison of spatial distributions of the 42 cloud classes with and without blurring patches. Notably, patches originally classified as low-cloud classes without blurring are now assigned to #5 and/or #13. I observe that some low cloud patches are falsely classified as #1, where they should be assigned as high-altitude cloud-dominated classes. This underscores the capability of RICC to capture fine-scale spatial structures when generating latent representations. Furthermore, stratocumulus decks are consistently classified as low cloud class #34, indicating that the blurring process does not impede the classification of specific types of stratocumulus clouds into the low cloud class, albeit at the expense of diversity in class assignments. Consequently, the capacity of the satellite emulator may necessitate enhancement to fully capture high-frequency information, given the influence of blurring on class assignments. The examination of the extent to which the blurriness of synthesized radiance affects the results suggests that synthesized patches retain information crucial for RICC to generate latent representations for low cloud patches, thereby preserving their classification as low cloud classes.

Overall, the results suggest that IFS clouds produce more low clouds compared to the

results from blurred MODIS patches, even though there is a performance limitation in synthesizing high-resolution MODIS radiance. I conclude that SCuBA framework tells that IFS produces excessive low clouds against the amount of low clouds captured by observation.

5.4 Future Work

Initial examination of SCuBA framework reveals the gap between climate models and observations by comparing their representations by self-supervised deep learning model. I plan further improvements to satellite emulator by adopting from probabilistic generative models for approximating fine cloud textures to fine-tuning climate foundation model for learning general mapping between climate variables and radiances.

The CNN-based satellite simulator shows its limitation in generating the high-frequency information in the structure. Since RICC from the real MODIS images learn significant texture information, satellite emulator needs to create higher-resolution images. I plan to use a generative adversarial network (GAN) and diffusion model. GAN is known to generate realistic high-resolution images. To mitigate the influence of blurred synthesized radiances, I use GAN to improve the quality so that the results of SCuBA from synthesized radiances may enrich their diversity of cloud classes. Diffusion model is an alternative approach to GAN given the unstable training due to the mode collapse. As diffusion model can generate realistic images that closely match the distribution of real MODIS radiance images, it may contribute to reduce blurriness in reconstructions. Conditional diffusion model is another methodology to address the issue by conditioning latent representations of lower-dimensional vectors that learn mapping from climate variables to individual MODIS bands.

I envision that fine-tuning a weather and climate FM for satellite emulator provides more generalizability: FM has capabilities of representing three-dimensional space and time of underlying physics, while a neural network, which is trained to learn a translation between input climate variables and output radiances, may not be fully grounded by physical laws. Thus, the use of FM for a satellite emulator can improve the interpretability of results instead

of training it from scratch.

CHAPTER 6

CLUSTERING AUTOENCODER

In this section, I describe the self-supervised deep clustering algorithm to perform mentality reduction and clustering. The work is done in internship and subsequent collaboration with Frontier Development Lab 2022 US program.

6.1 Related Work

Climate models and simulations can generate future projection simulation data, with increasing the spatial and temporal resolution as well as improving cloud-resolving radiative, and microphysics models [150, 9]. Future computing power eventually allows global climate models to simulate low clouds at 10m resolution until 2060 if computing power improves at a rate of the Moore's law [139]. At the same time, climate model simulation results are increasingly used for practical applications and more granular decision making such as climate resilience assessments at specific sites or for specific issues. For example, the impact of climate change on soil and groundwater contamination has been studied recently, because extreme precipitation and/or shifts in precipitation/evapotranspiration regimes could re-mobilize contaminants and proliferate contaminated groundwater [88, 82, 186]. This large volume of simulation outputs with high spatial-temporal resolution overwhelm the capacity of computing powers and resources when climate scientists and practitioners analyze trends and mechanism under warming climate scenarios. Furthermore, the rapid evaluation of climate impact assessment may downturn due to the increasing volume of climate simulations with a variety of ensemble of models and scenarios.

Climate classification – or identifying similar climatic regions or zones – has been used to understand the spatial variability of climate across a large area or facilitate the assessment of the climate change impact. Such classification essentially reduces the dimensionality of the vast climate simulation data into a set of zones. We can then understand the climate

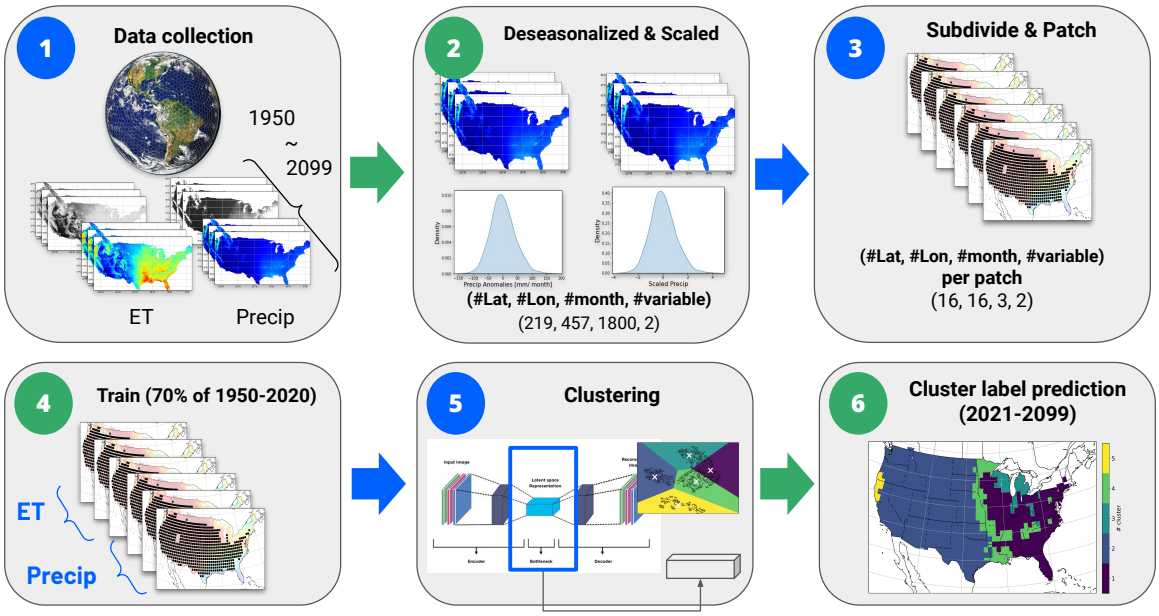


Figure 6.1: Illustration of our workflow of unsupervised climate data clustering.

spatiotemporal patterns at a particular region or location without querying large climate datasets. Historically, there are multiple climate classifications available across the world [71, 117, 25]. For example, the most popular and accepted climate classification, Köppen–Geiger classification (KGC) [70] well represents the empirical association with local vegetation and captures the long-term mean climatologies.

However, widely used climate classifications are often relied on deterministic definitions by human experts subjective based on prescribed thresholds [71, 117, 13, 25]. The Köppen-Geiger schema and the updated versions identify five main categories and 30 sub-categories based on the threshold values of temperature and precipitation. However, the schema is subjective to empirical biome distributions and thus difficult to scale when taking into account other variables [13, 25] for quantifying comprehensive similarities of climate patterns. In addition, there are some imperfections as different climate regions have been delineated based on the extent of plant species, rather than actual climatological parameters only temperature and precipitation [162]. At the same time, there is a need to address the uncertainties in the climate simulations. A simple average across ensembles of climate

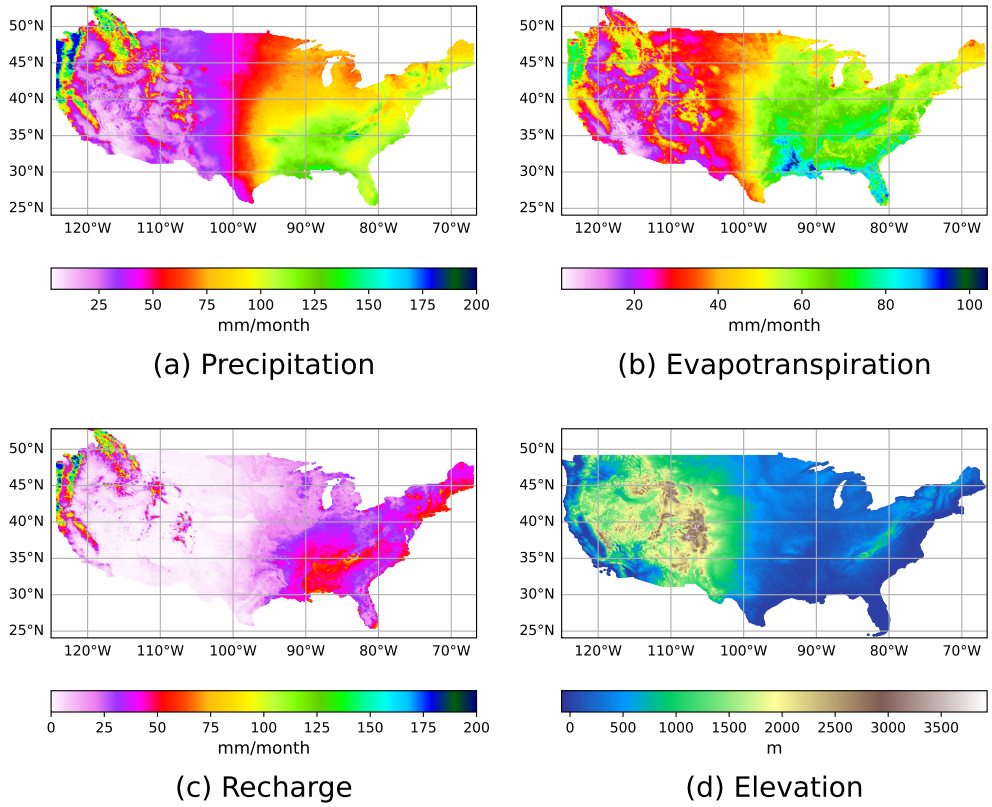
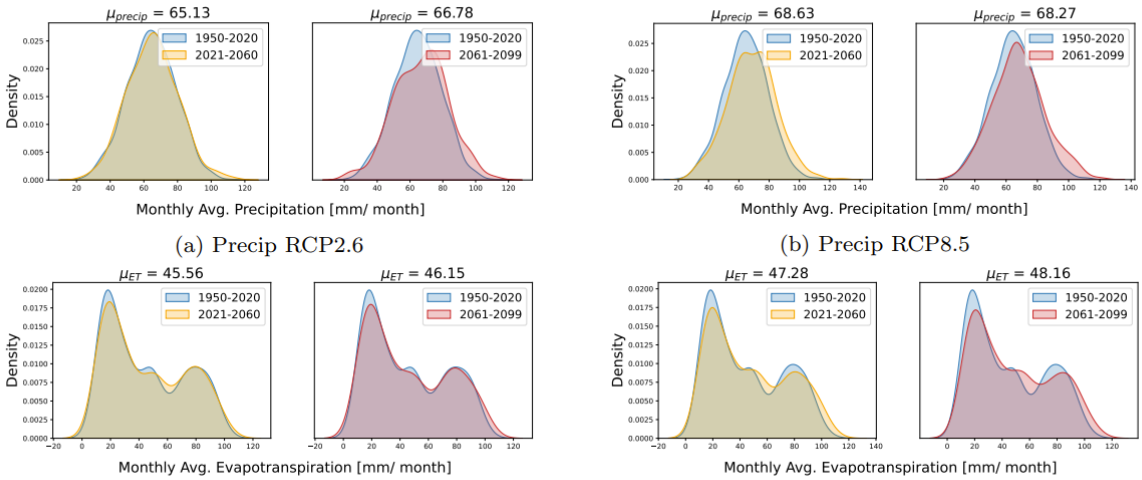


Figure 6.2: Four variables used for training and testing unsupervised climate clustering. (a) precipitation and (b) evapotranspiration are monthly averaged over from 1950 to 2099 from GFDL-ESM2G ; (c) recharge ratio is calculated based on the subtraction of the monthly evapotranspiration from the monthly precipitation, and then takes the average from 1950 to 2099; (d) elevation over CONUS uses PRISM 4km resolution dataset [120].

projections is insufficient to represent uncertainties of underlying physics in models by oversimplifying the transition of climate regimes both spatially and temporarily [37]. To address these limitations, it is important to develop a data-driven approach for climate zone delineation without subjective definitions. In particular, since climate change potentially alters these zones in the future, it is critical to develop objective and automated approaches for defining zones.

Machine learning (ML) techniques such as clustering have used new climate indices to better understand the complexity of weather and climate over the last decades. Clustering techniques are widely performed in Earth science to group various unique regional and global spaces based on weather/climate, hydro-climate, and ecohydrological patterns based on k-



(a) Precip RCP2.6

(b) Precip RCP8.5

(c) ET RCP2.6

(d) ET RCP8.5

Figure 6.3: Probability density function for the spatial averaged monthly precipitation and ET values of RCP2.6 and RCP8.5 scenarios from GFDL-ESM2G simulation over the entire CONUS area. We show the distribution from mid-century and late-century to be overlaid to that of historical data.

means clustering [196, 24, 169], hierarchical clustering [99, 29], multitask learning [114], and principal component analysis (PCA) [16]. (Note that PCA is also known as empirical orthogonal function (EOF) in climate science, and we use PCA and EOF interchangeably in this study.) Many of climate studies used PCA as a dimensionality reduction method [24, 22]. This approach has some limitations as it is used to find only a few of the strongest signals and all the signals are orthogonal to each other, which leads to difficult physical interpretation. While these unsupervised ML approaches characterize global and local unique patterns in data-driven fashion, there needs to be an exponential demanding computing power when an application size to clustering analysis has increased despite of available scalable clustering algorithms [10, 156].

Recently, artificial intelligence (AI) technologies including deep neural networks (DNNs) have evolved from a traditional rule-based to a data-driven approach, showing impressive classification, pattern recognition, and object generation skills in particular natural language processing [32, 185]. For climate science applications, DNNs have been increasingly used to

identify hidden patterns in climate studies from the large quantities of climate simulation datasets [17, 174]. For example, Liu et al. [84] first introduce DNNs to detect extreme climate patterns from historical records. Chattopadhyay et al. [22] leverages 4 layers of CNN to predict climate patterns over North America based on predefined 4 clusters via k-means. Mittermeier et al. [100] train DNNs with the Coupled Model Intercomparison Project 6 (CMIP6) large ensembles to classify 29 important circulation types over Europe. Autoencoders [72, 109] (AE), a deep learning technique that leverages dimensionality reduction, shows exploratory powers free from artificial assumptions to better represent underlying data structures [123, 75]. Tibau et al. [161] argue that using autoencoders as a dimensionality reduction method addresses some problems the classical methods exhibit: firstly, finding the proper kernel to keep a nonlinear structure of data into a low-dimensional space and secondly, creating understandable latent space (representation of original data in the lower-dimensional space) compared to the standard methods. Furthermore, as the typical dimensionality reduction approach usually involves summarizing climate model output information by calculating means and variances without maintaining the information about extreme weather events, the advanced DNN approach may open up a new research approach to address this problem.

In this study, we present an unsupervised climate classification workflow to reduce the dimensionality of complex spatio-temporal climate simulation outputs, and to identify and map distinct climatic zones across the continental US (CONUS) and their changes in the future. In particular, we leverage autoencoders to reduce the dimensionality of the extensive climate data on lower-dimensional space that captures essential climatological information. We also develop a clustering autoencoder for climate classification by integrating an online clustering algorithm to reduce exponentially increasing clustering time and increase the capacity of data for further generalization. Although this study focuses on the precipitation and evapotranspiration (ET) that are relevant to hydrological impacts such as droughts and groundwater assessments, the approach is general for any climate variables, supporting

access to complex climate simulation outputs in a compact form by keeping the relevant climate information. All the workflow is based on the cloud platform such that we can assess the climate classification anywhere across the CONUS without downloading vast amounts of climate data.

6.2 Data Collection

Geophysical Fluid Dynamics Laboratory Earth System Model (GFDL-ESM2G) [34, 35] is one of the participated models in CMIP5 [157] that provides a comprehensive ensemble simulation framework of global climate projections under different represented concentration pathway (RCP) scenarios. In this study, we use the downscaled monthly dataset [96]. The ocean dynamics of GFDL-ESM2G highlights the high fidelity of ocean carbon and heat content variability, resulting in a relatively shallow thermocline and weaker ENSO compared to observations. We select on the GFDL-ESM2G to train and test our unsupervised learning approach due to the less biased for the precipitation and temperature (i.e., associated with evapotranspiration) projections [66].

In particular, our study focuses on the CONUS region that covers a spatial extent of 25.5625° N to 52.8125° N, −124.0625° E to −67.0625° E. The dataset we used has the grid cells that contain mean-monthly climatic variables downscaled to a fine resolution of 0.125° using monthly Bias-Correction Spatial Disaggregation (BCSD) techniques [127]. The covered area contains 219 × 457 grid cells. Used data are based on global climate projections from the World Climate Research Programme's CMIP 5. The BCSD procedure is done in two steps: Firstly, Global Climate Model (GCM) historical simulations are compared to observations in order to identify and remove biases from the projection dataset using quantile mapping (a recent bias-correction approach, popular in climate science [159]) constructed from daily GCM simulations and observation values. Secondly, the GCM projections of step 1 are spatially downscaled to the desired resolution. It is performed for the whole spatial domain on a specific timestep basis. Finally, in this step, the historical climatology and spatially

disaggregated changes of the given timestep measured from that climatology are merged. The downscaled dataset is publicly available [96].

Since precipitation and evapotranspiration have different ranges of values, we standardized data in order to provide an equal contribution of each variable to our cluster analyses (see step 2 in Fig. 6.1). Standardization is performed for each grid cell by removing the mean and dividing by the standard deviation. In general, a single or combination of climate variables, are defined as the anomaly time series to accurately describe climate variability over large areas than the raw data would do. Moreover, the nature of climate variables is to contain recurrence patterns of seasonality, especially dominant in the mid-latitude regions, leading to strong temporal autocorrelations. To get rid of the seasonal component, we removed the three-month-running mean from the monthly data of both, precipitation and ET. Finally, we applied area weighting for latitude to each grid cell for each monthly value to ensure that the value of each latitude and longitude location is treated equally. We achieve this by weighting each data by the cosine of the latitude. Given that the outputs of climate models are uncertain and the projected changes of precipitation events are not homogeneous in space and time, in this study, we incorporate all four Representative Concentration Pathway (RCP) scenarios (i.e., RCP 2.6, 4.5, 6.0, 8.5) from GFDL-ESM2G that participates in the CMIP5.

After the transformation of data, we then handle each snapshot at 219×457 grid cells as image data to enable efficient learning of physical features by deep neural networks. We spatially subdivide 219×457 pixels "image" data into a 16×16 pixels $\times$ 3 month scale, $\approx 2^\circ \times 2^\circ$ area, giving a smaller geographical and temporal unit, *patch*. The patch creation process (i.e., regrid of climate snapshot image) is performed by sliding every 8 grids spatially with the extraction of large numbers of overlapping patches. This may provide an additional degree of translation invariance to our neural networks. We then split the patches into three windows: *historical* (1950–2020), *mid-century* (2021–2060), and *late-century* (2061–2099). We only use 1 468 214 patches sampled from 70% of patches at the historical time window

to train our neural networks and leave 30% for clustering and testing.

In addition to the climate variables, we also consider elevation data at 4 km spatial resolution from PRISM dataset and regrieded to 14 km [120] to compare resulting cluster patterns from autoencoder and clustering. Fig. 6.2 visualizes long-term monthly mean ((a)–(c)) for precipitation, ET, and recharge (difference between precipitation and ET), as well as elevation data. We add elevation to our evaluation because it is roughly associated with precipitation and temperature variables used by the KGC classification scheme [71].

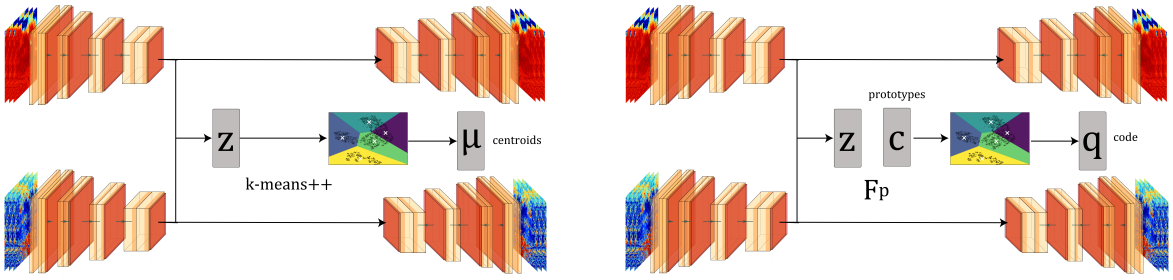
Fig. 6.3 presents the probability density functions (PDF) of precipitation (a, b) and ET (c, d) for RCP 2.6 (left column) and RCP 8.5 (right column) to investigate differences between two RCP scenarios for the entire dataset. We highlight the lowest and highest radiative forcing scenarios used in the fifth Intergovernmental Panel on Climate Change assessment report. Each plot shows distributions of monthly spatial averaged data across the CONUS. There is a historical PDF marked in blue against mid-century or late-century PDFs in orange and red, respectively. We also calculated the mean monthly precipitation and ET for each future time window. Different mean monthly values are noticeable between the three time periods.

As shown in Fig. 6.3, the distributions of precipitation and ET in the mid and late-century shift towards higher than the values in historical time and become the long tail in extreme events. The value of mean precipitation during the mid-century ranges between 65.13 and 68.63 mm/month for RCP 2.6 and RCP 8.5 respectively, and ranges between 66.78 and 68.27 mm/month during the late century. Similarly, the value of mean ET increases from 45.56 and 47.28 mm/month to 46.15 and 48.16 during the late century for RCP 2.6 and RCP 8.5. Moreover, it is important to notice that the most differences between analyzed models leaned toward extreme weather events. As it is known, extreme events are expected to be more severe in the future due to climate change which will have significant impacts on buildings and infrastructure, as well as groundwater flow and contamination transport. Since the PDFs indicate more frequent extreme events in future projections, we expect that

our resulting clusters via autoencoders (see Section 6.3) may capture different patterns and frequencies.

6.3 Methodology

Unsupervised climate data clustering shown in Fig. 6.1 serves to reduce the dimensionality of climate simulation outputs by convolutional autoencoders and to group the lower-dimensional representation (i.e., *latent representation*) into similar climate patterns by clustering techniques. The resulting 512 dimensions at the latent representation can approximate a 666 176 × reduction of comprehensive climate information at 219 × 457 grid cells × 71 years from 1950 to 2020 × 4 RCP scenarios. Our workflow is composed of five elements: 1) *Download CMIP5 simulation data* as a source of historical and future projection products, from which we extract downscaled precipitation and evapotranspiration variables that determine the net flow of the groundwater system over CONUS; 2) *Transform the downloaded data* through deseasonalizing and scaling; 3) *Subdivide data and Patch creation* to generate a smaller geographical unit of data to be efficient learning (see Section 6.2 for step 1–3); 4) *Train the convolutional autoencoders* to reduce the dimensionality and then to produce the latent representation; 5) *Apply clustering* to the latent representation for grouping the pixels into similar types of climate patterns. In particular, we develop and compare the two types of autoencoders : A) Train standard convolutional autoencoders and k–means algorithm separately, naming *standard autoencoder*. B) Develop a joint loss function to train convolutional autoencoders and online clustering algorithm simultaneously, giving *clustering autoencoder*. This clustering autoencoder combines step 4 and 5 in a same training process to be more scalable for a larger set of data in assigning clusters. We describe our two algorithms in turn; and 6) *Cluster label prediction* step finally assigns cluster labels to all future climate projection data so that we can evaluate trends in the distribution of different climate patterns.



(a) Standard autoencoders (Section 6.3.1) (b) Clustering autoencoder (Section 6.3.2)

Figure 6.4: Diagrams of two autoencoders.

6.3.1 Standard autoencoder

The autoencoder [72, 109] is a commonly accepted unsupervised learning algorithm to map important information in input images x into latent representations z through dimensionality-reduction and then reconstructs the original input image from the representations as the output. Training of autoencoder minimizes differences between the input images and their output images through the encoder E and decoder D.

Given a set of input images $X = \{x_1, \cdots, x_n\}$ that are encoded with encoder as $Z = E(X)$, their reconstructed images are $\hat{X} = D(Z) = D(E(X))$. Loss function $\mathcal{L}(\theta)$ quantifies the difference between X and $\hat{X}$ as following:

$$\mathcal{L}(\theta, \phi) = \sum_{x \in S} ||x - D_\theta \left(E_\phi(x) \right)||_p^p, \tag{6.1}$$

where S is a set of training images; (ϕ, θ) represents trainable parameters in encoder and decoder, respectively, and $||\cdot||_p^p$ is the pth power of the p-norm of the inputs and restorations. We specify $p = 2$ to calculate the L_2 loss throughout this study. That is, the optimization of Eq. 6.1 minimizes the difference between input and output images so that the autoencoder eventually generates high-fidelity reconstructions.

The performance of image recognition improves with multiple layers of convolutional filters [147] by extracting useful representations through the stack of these nonlinear filters.

Therefore, we integrate convolutional layers into *block*, each incorporating the same size of two convolutional layers, and design a symmetric encoder-decoder structure (see Fig. 6.4a and Tables 6.1, 6.2).

Table 6.1: Encoder architecture. The table shows the names of layers, the shape of the tensor, and the number of parameters at each row. 'Conv2d' denotes convolutional 2d;'ReLU' denotes a rectified linear layer activation; 'Batch norm' denotes batch normalization. Shpae represents the minibatch size (#B), height, width, and channel. The number of total trainable parameters in encoder is 126 800.

Layer	Shape	Parameter
Input	(#B, 16, 16, 3)	0
Conv2d/ReLU	(#B, 16, 16, 16)	1216
Conv2d/ReLU	(#B, 16, 16, 16)	6416
Conv2d/ReLU	(#B, 8, 8, 32)	12832
Batch Norm	(#B, 8, 8, 32)	128
Conv2d/ReLU	(#B, 8, 8, 32)	25632
Conv2d/ReLU	(#B, 4, 4, 64)	18496
Batch Norm	(#B, 4, 4, 64)	256
Conv2d/ReLU	(#B, 4, 4, 64)	36928
Conv2d/ReLU	(#B, 2, 2, 128)	8320
Batch Norm	(#B, 2, 2, 128)	512
Conv2d/ReLU	(#B, 2, 2, 128)	16512

Table 6.2: Decoder architecture. 'Conv2d Transpose' denotes a transposed convolutional 2d operation, and other operations are the same as encoder in Table 6.1. The number of total trainable parameters in decoder is 392 563.

Layer	Shape	Parameter
Conv2d Transpose/ReLU	(#B, 4, 4, 64)	204864
Conv2d/ReLU	(#B, 4, 4, 64)	102464
Batch Norm	(#B, 4, 4, 64)	256
Conv2d Transpose/ReLU	(#B, 8, 8, 32)	51232
Conv2d/ReLU	(#B, 8, 8, 32)	25632
Batch Norm	(#B, 8, 8, 32)	128
Conv2d Transpose/ReLU	(#B, 16, 16, 16)	4624
Conv2d/ReLU	(#B, 16, 16, 16)	2320
Batch Norm	(#B, 16, 16, 16)	64
Conv2d/ReLU	(#B, 16, 16, 3)	1203
Output	(#B, 16, 16, 3)	0

Each block has the same size of kernel but decreases the size from 5, 3, and then 2 based

on our hyperparameter search. We add batch normalization [55] to enable a stable and faster training process. We train different autoencoders for precipitation and ET respectively on 1 468 214 patches. Because our empirical evaluation shows that a better training performance gains during the training of autoencoder for each variable separately rather than training both variables as one input image.

Having a trained standard autoencoder, we cluster the latent representation to identify unique climate patterns. We apply k-means++ [5] as known for the probabilistic initialization to find an initial seed of k number of clusters, and the approach outperforms the native k-means algorithm. In implementation, we use k–means++ API provided by scikit-learn Python package [116] to 630 166 historical patches unseen in training of autoencoders. We separate the dataset for clustering from the training dataset because k-means++ has a memory limitation to fit all our training patches. We then obtain a set of k cluster centroids, $\mu = \{\mu_1, \cdots, \mu_k\}$. We determine the optimal number of clusters via the elbow method [15], a heuristic approach used in determining the optimal number of clusters. See Section 6.4.1 for the result.

6.3.2 Clustering autoencoder

The separate training of autoencoder and clustering can be a computational bottleneck in terms of both time and resources especially when the amount of dataset becomes large. Given a dataset of N number of patches and D number of dimensions to apply k-means clustering with k number of clusters, the memory space complexity is $N(D+k)$, which could be impractical for large N. While scalable clustering algorithms are widely available [10], a clustering autoencoder (see Fig. 6.4a) and clustering training can further benefit the scalability. A joint loss function [3] formulates a combination of both reconstruction and clustering loss terms:

$$\mathcal{L}_{\text{joint}}(\phi, \theta) = \lambda_{\text{reconst}}\mathcal{L}_{\text{reconst}}(\phi, \theta) + \lambda_{\text{clustering}}\mathcal{L}_{\text{clustering}}(\phi), \tag{6.2}$$

where $\mathcal{L}_{\text{reconst}}$ corresponds to L_2 loss that quantifies the difference between a training data x and output of autoencoder $D_\theta(E_\phi(x))$; $\mathcal{L}_{\text{clustering}}$ term learns clustering assignments through an online approach; two coefficients λ_{reconst} and $\lambda_{\text{clustering}}$ balance the two terms to achieve smoother optimization and better representations. We choose $(\lambda_{\text{reconst}}, \lambda_{\text{clustering}}) = (0.6, 0.4)$ in this study as the combination leads to the minimal loss values. The learned representation via the clustering approach reflects the association of group features of clusters by optimizing the joint loss function with Eq. 6.2.

Our $\mathcal{L}_{\text{clustering}}$ is motivated by an online method [19] with simplifying the cross entropy loss between two cluster assignment: "codes" q_c via the Sinkhorn-Knopp algorithm [27] and p_c that is computed as "prediction" obtained with a softmax of the dot product of K trainable prototype $C = \{c_1, \cdots, c_K\}$ and the latent representation $z_c = E_\phi(x)$. We calculate the dot product with an output layer from a single layer perceptron F_p such that $z_c^\top c = F_p(E_\phi(x))$. Thus the second loss term in Eq. 6.2 is

$$\begin{aligned} \mathcal{L}_{\text{clustering}}(\phi) &= -\sum_{k \in K} q_c^{(k)} \log\left(p_c\right) \\ \text{where} \quad p_c^{(k)} &= \frac{\exp\left(\frac{1}{\tau} z_c^\top c_k\right)}{\sum_{k'} \exp\left(\frac{1}{\tau} z_c^\top c_{k'}\right)}. \end{aligned} \tag{6.3}$$

A temperature parameter τ [182] adjusts the probability distributions. For example, a smaller τ adds more probability to the largest value of $z_c^\top c_k'$, in opposite, a larger τ penalizes them more evenly. In this study, we choose $\tau = 0.1$ as following the standard practice.

Along with the optimization, given the B feature vectors, the set of codes $q_c = \{q_c^1, \cdots, q_c^B\}$ is defined as

$$q_c = \text{Diag}(u) \exp\left(\frac{C^\top Z}{\epsilon}\right) \text{Diag}(v), \tag{6.4}$$

where u and v are computed by renormalization in $\mathbb{R}^K$ and $\mathbb{R}^B$ respectively through iterative Sinkhorn-Knopp algorithm. ϵ is a smoothing parameter. We set $\epsilon = 0.115$, which achieves similar physical regimes and cluster patterns to ones from standard autoencoder (see Section 6.4

in detail).

6.3.3 Training scheme

We train two convolutional autoencoders in Fig. 6.4 by stochastic gradient descent [79] as an optimizer with a learning rate 10^{-2} on four NVIDIA K80 GPUs, an NVIDIA V100 GPU, and an NVIDIA P100 GPU based on machine availability on the Google Cloud Platform. We train convolutional autoencoder over 1 468 214 patches until 200 epochs for standard autoencoder and until 400 epochs for clustering autoencoder with minibatch size 1024. To gain an acceleration of multi-GPUs instance environment, we use horovod [143]. Note that we use GPU for training and CPU for the inference/prediction step for both autoencoders.

6.4 Evaluation

To evaluate differences in how climate patterns change through time, note again that we split the data into three windows: historical (1950–2020), mid-century (2021–2060), and late-century (2061–2099). Such time windows are necessary to quantify the changes over this century because the inter-annual variability is quite large compared to the overall trend over the 100 years.

While our primary target is precipitation and ET, we examine the difference of resulting climate patterns from recharge rate, which subtracts ET values from precipitation (i.e., a net intake flow to the groundwater system). These two quantities are critical for water resource-related questions including soil and groundwater contamination. In addition, we used elevation as an indicator of topography and a proxy of air temperature variable and temperature as well, which are often associated with both precipitation and ET to quantify the influence from different combinations of variables. Thus, for the climate classification process, we group the results from the autoencoder into four parts:

- train autoencoders and clustering separately on precipitation and ET;

- train autoencoder and clustering on recharge rate;
- train autoencoder and clustering on recharge and elevation;
- train a clustering autoencoder on precipitation and ET values.

For all analyses in this section, we work with 1 476 425 test patches that have not been included in the training stage.

6.4.1 Optimal number of clusters

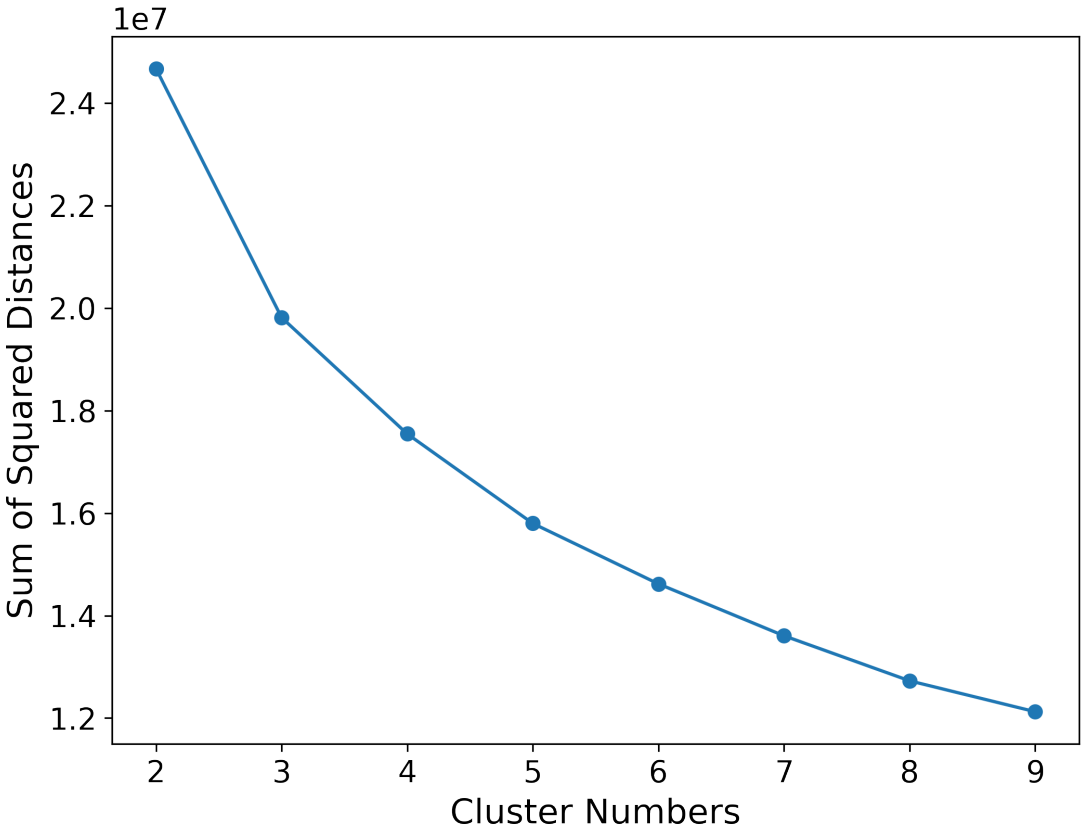


Figure 6.5: We plot the sum of squared distances between patches used in k-means clustering and resulting centroids as a function of the number of clusters. We observe that the sum of squared distance decreases almost linearly at 5 clusters, and for this reason, we choose 5 clusters as our optimal number of clusters.

First we determine an optimal number of clusters. In this study, we test a range of $k \in \{2, \cdots, 9\}$ and elbow method indicates that five clusters are our optimal number of clusters, which is a pivot point in our study where the sum of square distance among latent representations resulting from patches in the historical time window (see Fig. 6.5). The results suggest that at least five clusters need to characterize the representative patterns in the dataset. We left the discussion of the physically optimal number of clusters for future work. For the rest of our study, we present clustering results working with five clusters.

6.4.2 Runtime performance experiment

We perform a scaling experiment to examine the efficiency of the clustering process by clustering autoencoder against k–means algorithm used for the clustering step in standard autoencoder, described in Section 6.3.1. The experiment setup is to measure the completion time of clustering runtime taken by each approach on a single CPU. We scale a range of sizes of patches $\in \{10, 100, 1000, 10000, 100000, 100000, 500000, 1000000\}$. This study defines clustering runtime as k-means process time for standard autoencoder and a neural network prediction time for clustering autoencoder.

The scaling experiment result is shown in Fig. 6.6. We observe that the completion time by k–means clustering exponentially increases with the number of application sizes, whereas the execution time by clustering autoencoder linearly increases, and the highest performance result by clustering autoencoder gains 9.23 times faster for a case of 1 000 000 patches. The result emphasizes that the clustering autoencoder can show significant scalability when applied to larger sizes of climate datasets.

6.4.3 Learned representation

We conduct a qualitative test of latent representations to assess their structure and quality because our climate clusters are derived from clustering results applied to them. We use t-distributed Stochastic Neighbor Embedding (t-SNE) [163], which projects higher-

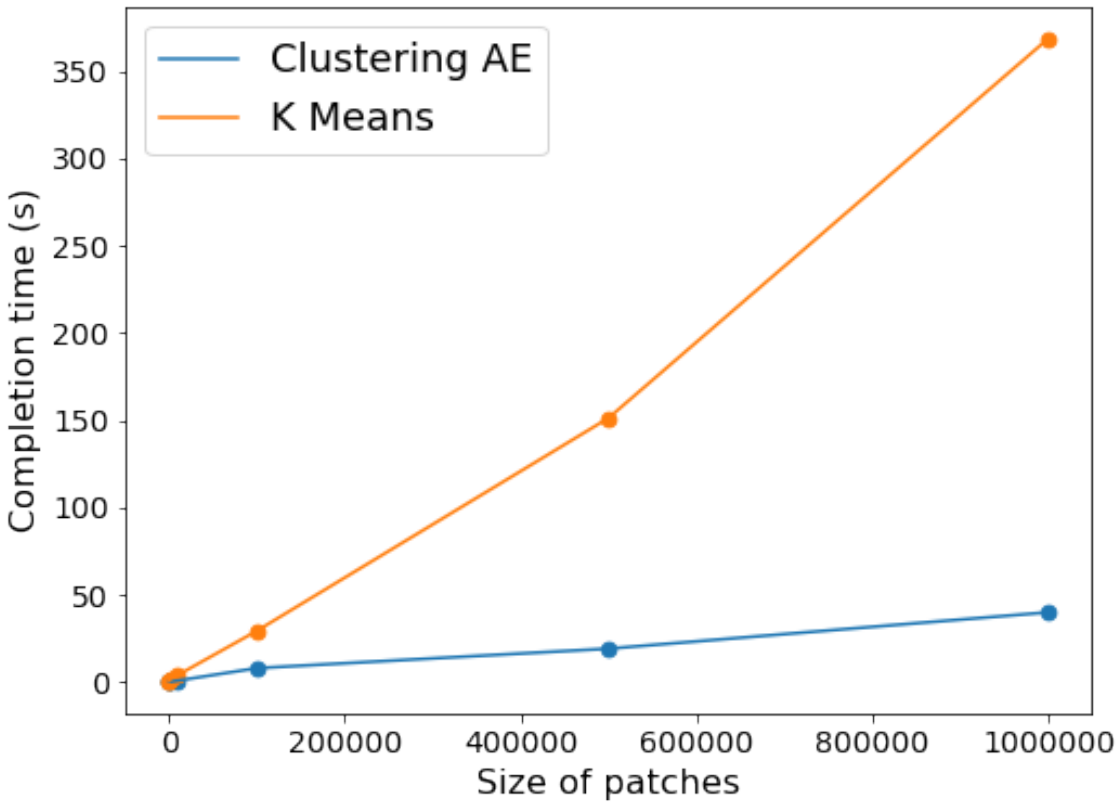


Figure 6.6: Scaling results in terms of completion time (second) with clustering by clustering autoencoder (blue) and k–means clustering to latent representations from each autoencoder. The dots represent the completion time as a function of the number of patches being clustered by both algorithms. We test the size of a set of patches from 10 to 1 000 000. Clustering autoencoder shows significant advantages to reduce computation time for clustering in particular for the larger size of applications, indicating that the algorithms can scale efficiently to work with a larger amount of climate simulation datasets.

dimensional data onto a 2-D map mostly used for visualization of a high-dimensional data structure, to examine whether the spatial structure of latent representations produced from our two autoencoders captures meaningful association with physical variables. t-SNE keeps the structure of data in a high-dimensional space at a projected space (often 2-D) in such a way that a pair of similar representations of data is projected near to each other, while dissimilar pairs are in distant positions.

We then apply t-SNE to the latent representations from both autoencoders for 2000

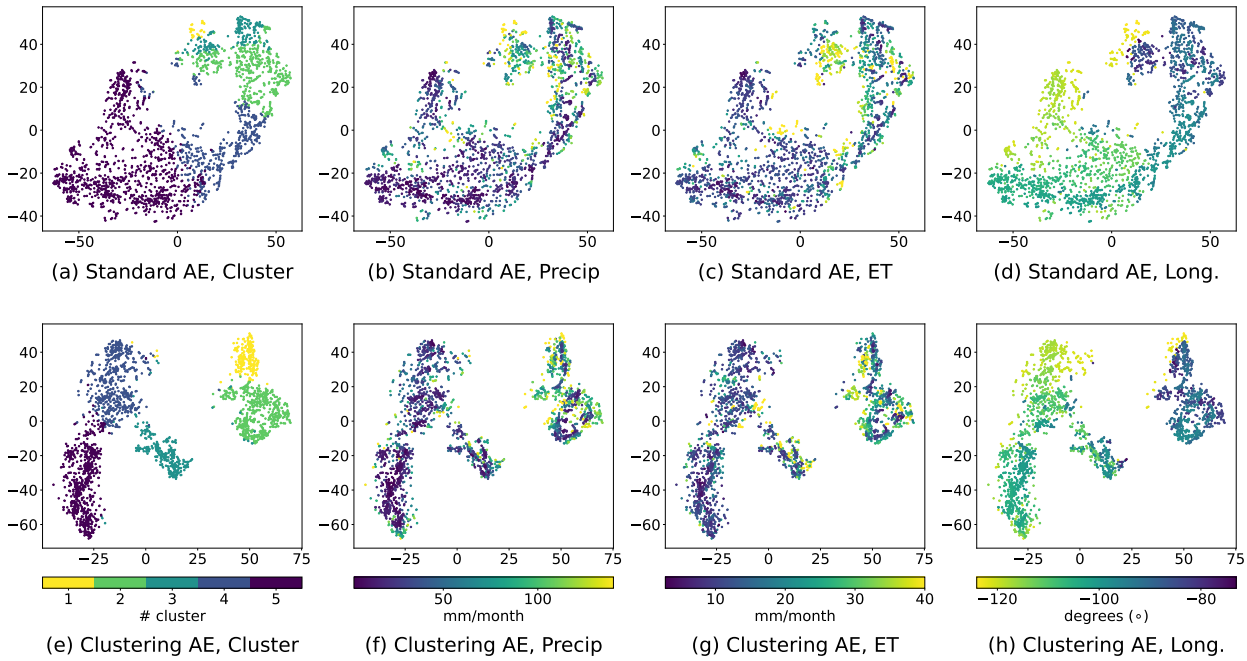


Figure 6.7: t-SNE visualization of the latent representation of 2000 test patches from (a)–(d) standard autoencoder in Fig. 6.4a and (e)–(h) clustering autoencoder in Fig. 6.4b. Patches are randomly selected from a set of test patches unseen in the training stage. Each patch is colored by cluster assignments (a) and (e) (see Section 6.4.1 for determining an optimal number of clusters), patch-wise mean precipitation values (b) and (f), patch-wise mean ET values (c) and (g), and longitude values (d) and (h) respectively. We label cluster numbers in order of within-cluster mean precipitation value in descending order (i.e., #1 has the highest precipitation mean, in contrast, #5 has the lowest). We observe that the structure of latent representations, cluster assignments, and physical variables are not randomly projected on the map, indicating that the latent representations generated from both standard and clustering autoencoders capture meaningful aspects of physical properties.

patches sampled at random from the test dataset. Fig. 6.7 visualizes 2-D t-SNE projections where each color represents cluster numbers (leftmost column), patch-mean precipitation (left column), patch-mean ET (right column), and patch-center longitude values (rightmost column) for standard autoencoder at upper panels and for clustering autoencoder at lower panels respectively. Note that, we assign cluster numbers based on within-cluster mean precipitation in descending order. We see in Fig. 6.7 (a) and (e) that patches in the same cluster locate coherently and those in different clusters are put far away, suggesting that autoencoders achieve learning separable latent representation among dissimilar patches. We also observe that the latent structure from the clustering autoencoder shows clearer

inter-cluster segregation, indicating that the training does not fall into a trivial solution (i.e., all latent representations become identical) during optimizing the additional cross entropy term 6.3. t-SNE panels colored by precipitation and ET highlight that similar physical properties are adjacent in the projected maps. Because our autoencoders work with precipitation and evapotranspiration data that exhibit a longitudinal subdivision in Fig. 6.2, the resulting latent representations in Fig. 6.7 (d) and (h) capture the association with longitudinal values. These results suggest that our network learns underlying patterns within input data. In summary, we conclude that our autoencoder can capture nonlinear physical relationships and then generate non-trivial latent representations.

6.4.4 Physical regimes

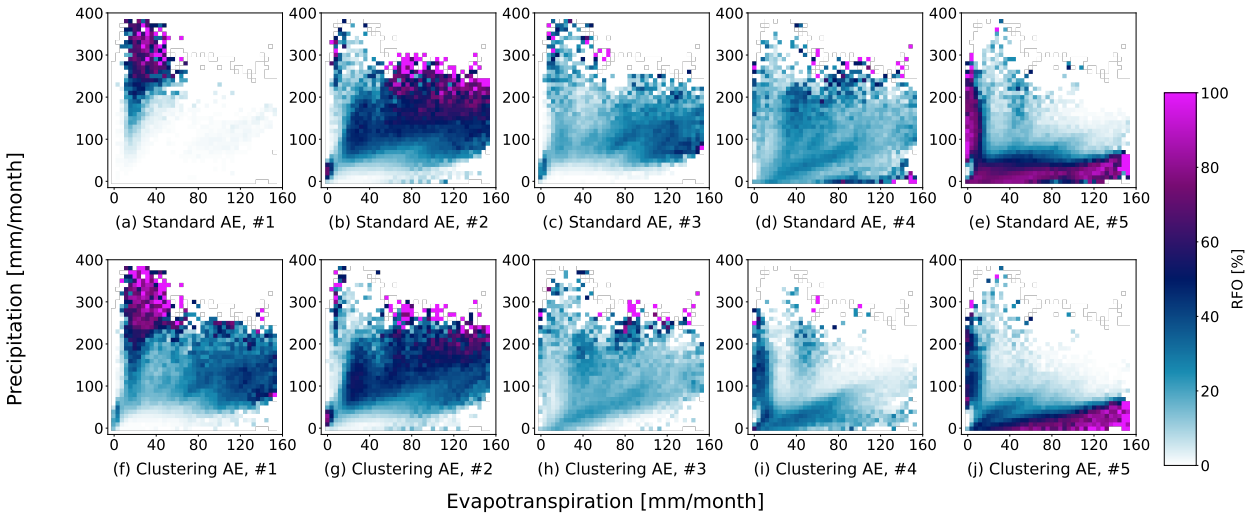


Figure 6.8: Heatmap histograms of the relative frequency of occurrence (RFO) on a joint histogram of evapotranspiration (ET) – precipitation (PR) space to the distribution of patches in RCP 8.5 from (a) standard autoencoder (Fig. 6.4a) and (b) clustering autoencoder (Fig. 6.4b). Panels (a) and (b) show the within-patch mean ET and precipitation values in bins of 4 mm/month in ET and 10 mm/month in precipitation. Both autoencoders generate a clear cluster partition, where #1 is dominant in domains at approximately ET $<$ 80 mm/month and precipitation $>$ 100 mm/month, #2 is dominant in higher ET and precipitation, and #5 is dominant in domains at ET $>$ 40 mm/month and precipitation $<$ 100 mm/month. The results suggest that autoencoders capture distinct physical features and reflect them into latent representations, giving unique climate clusters in the physical space.

We first evaluate whether climate clusters produced by our two autoencoder algorithms have reasonable physical associations. That is, resulting clusters should identify unique physical regimes among clusters on a two-dimensional joint ET–precipitation histogram. We calculate the relative frequency of occurrence (RFO) of clusters from one of five clusters (see Section 6.4.1) whose average ET and precipitation fall at a bin defined by every 4 mm/month in ET and every 10 mm/month in precipitation across the same two-dimensional ET–precipitation space. Thus, 100% of RFO means that a bin is only classified into one of five clusters. We plot the RFOs of patches from each cluster on the ET–precipitation space(Fig. 6.8) for RCP 8.5. Note that, we sort the numbers on the within-cluster mean precipitation in assigning cluster labels. Since the clustering algorithm assigns cluster number labels at random, we sort the cluster numbers in descending order of the mean precipitation values: the smallest number, #1 (hereafter we use # to depict cluster number) has the largest value of mean precipitation (wettest) and in contrast, #5 has the least value of mean precipitation (driest).

We see distinct unique physical regimes in clusters from both autoencoder approaches. Fig. 6.8 tells that for both autoencoders #1 is a combination of high precipitation (200–400 mm/month) and low ET (20–60 mm/month), while the #1 from clustering autoencoder could contain even large ET (80–150 mm/month) values and medium precipitation (70–250 mm/month). Similarly, #2 (Fig. 6.8(b) and (g)) from both autoencoders represents the identical physical regime that is distributed at relatively high precipitation (200–300 mm/month) and high ET (60–160 mm/month). In contrast, cluster #5 is characterized by either extremely low precipitation in a combination with all ranges of ET values or low ET with the combination of all ranges of precipitation values for both autoencoders. #5 from standard autoencoder (Fig. 6.8(e)) has the L-shape distribution that put together the two different climate situations in one regime, whereas #5 from clustering autoencoder (Fig. 6.8(f)) mainly helps to group data with ET ranging from 40 mm/month to over 160 mm/month under conditions of low precipitation. This allows for the separation of

situations with extremely low ET and a wide range of precipitation by merging them into cluster #4. Because Fig. 6.7 explains that the additional loss term in Eq. 6.2 for clustering autoencoder encourages latent representation to generate clearer boundaries among similar data. The other clusters #3 and #4 from both autoencoders fill in the intermediate physical regimes. As will show it later, #4 of the standard autoencoder spatially overlaps with #3 of the clustering autoencoder. Interestingly, the main physical regimes, which are commonly seen from both autoencoders, indicate that features extracted by autoencoder may capture representative patterns in climate data, and differences seen among the five regimes account for the combination of a loss function and clustering techniques. In summary, our five clusters are physically reasonable by distinguishing different physical regimes among clusters, supporting that a data-driven approach can provide rich information on climate patterns rather than performing a simple threshold approach.

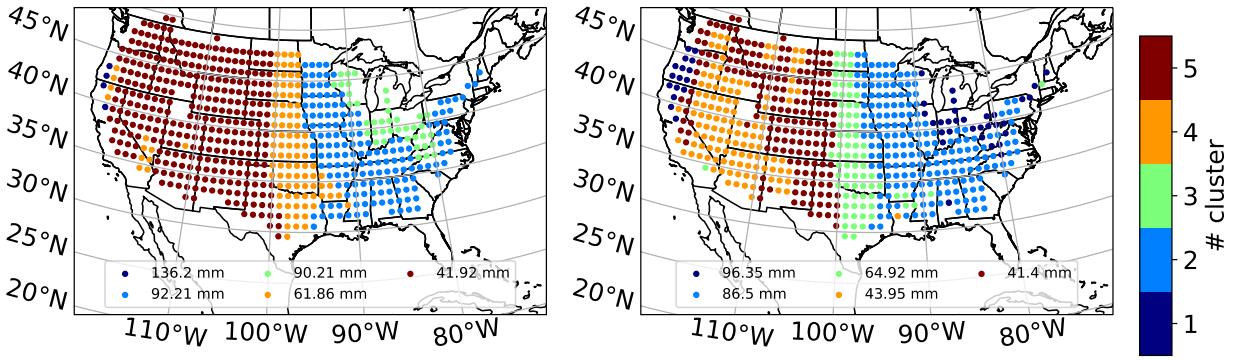


Figure 6.9: Spatial distribution of the most frequent one of 5 clusters generated from (a) standard autoencoder and k-means clustering and (b) clustering autoencoder. Dot points represent the center of patch, $2^\circ \times 2^\circ$ area. Patches are overlapped 1°, dot points are 1° resolution. The cluster labels colored by cluster number are assigned in descending order of within-cluster mean precipitation, and legend shows the mean values. Clusters in (a) and (b) show uniform geographical patterns within a same cluster but exclusive among different clusters, and those patterns are roughly matched with KGC's five climate classes: #2 corresponds with C (temperate) climate as well as #4 and #5 correspond with B (dry) climate. Yet, standard and clustering autoencoders capture a unique climate pattern in #1, suggesting the exploratory power of a data-driven approach.

6.4.5 Cluster patterns for precipitation and ET via standard autoencoder vs clustering autoencoder

In this section, we compare the results of spatial distributions of clusters from the standard autoencoder and clustering autoencoder for each of which we calculate the most frequent cluster as well as the relative frequency of occurrence of clusters at each patch location. To examine whether our climate clusters exhibit unique spatial patterns over CONUS, we analyze spatial patterns for different time windows. Note again that, in the training stage, autoencoders do not learn time and location information explicitly.

Fig. 6.9 shows the distribution of clusters for the late-century time window via standard autoencoder and clustering autoencoder. Again, in both cases, we sort the cluster numbers in descending order of the mean precipitation values: the smallest cluster number, #1 has the largest value of mean precipitation and in contrast, #5 has the least value of mean precipitation. We observe that the spatial patterns for both approaches do not excessively change over different times under the warming scenario among all three cases, while within-cluster mean precipitation increases compared to the values from 1950 to 2020 (see Fig. 6.10 for more details). The average percentage of five clusters over one of the three-time windows alters at most 1% of patch locations, whereas the monthly change shows at most 8% — the 8% change can still not occur every month. The few minor transitions in cluster positions from both autoencoders appear at the boundaries of clusters in geographical space. For instance, we may notice that the few patches over southeast California and southern Nevada change cluster number from #5 to #4 from the historical to the late-century time window, meaning that the region is projected to get wetter by the end of the century. But, a few patches over southwestern Oregon and northwestern California are changing cluster numbers from #1 to #4, meaning the region is projected to get drier by the end of the century. Similarly, clusters alter to wetter clusters in eastern Texas and western Louisiana as this region gets more precipitated over time. The results suggest that our climate clusters rather capture the consistent climatological distinctions over CONUS. It is interesting to note that both,

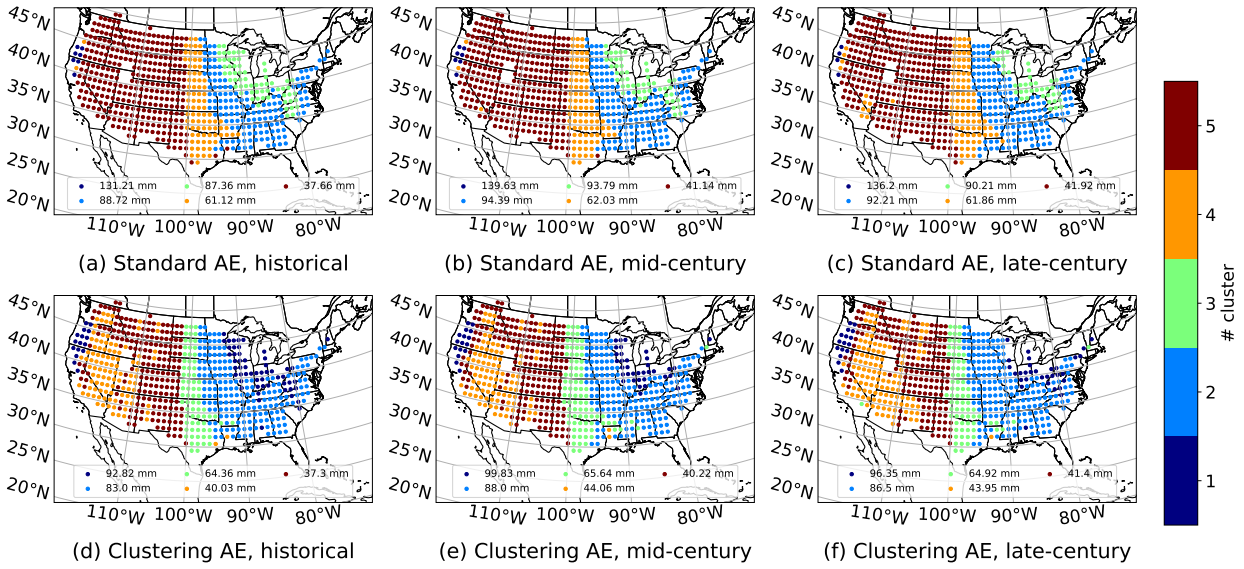


Figure 6.10: Spatial distribution of clusters from both standard and clustering autoencoder over three time windows. The upper panels are results from standard and the bottom ones are from clustering autoencoder. We here depict only results from RCP 8.5 scenario. We observe that the central mid-west region is expected to get drier (i.e., the most frequent cluster changes from #1 to #2) by the end of the century. The location difference of cluster #1 between two autoencoders is reasonable as the #1 from the clustering autoencoder has a considerably smaller mean precipitation value than the #1 from the standard autoencoder. It is likely that standard autoencoder groups only the western part of the CONUS into one cluster.

standard and clustering autoencoders, give higher within-cluster precipitation mean during the mid-century than the means during the late-century cluster.

As expected that two different autoencoders produce equivalent physical regimes in Fig. 6.8 at physical space, we also observe similar patterns of spatial distributions of clusters in Fig. 6.9: the driest and relatively drier clusters (#5 and #4, respectively) are likely assigned at a Rocky Mountain region, and the eastern part of the South and North Great Plains, where the wettest cluster (#1) is located at the far west coast of CONUS. Cluster #4 from the standard autoencoder and cluster #3 from the clustering autoencoder are qualitatively and quantitatively comparable by the location as well as by the means of precipitation (61.86 mm and 64.92 mm, respectively). We highlight the spatial similarity of individual clusters by investigating RFOs of each cluster at each patch location in Fig. 6.11 whether clusters are

spatially localized or widely spread. Spatial RFOs shown for each cluster are visually similar. Clusters #2 and #5 roughly match their spatial domain, #4 from standard autoencoder and #3 from clustering autoencoder overlap each other, while #1 is different between two autoencoders such that #1 in clustering autoencoder combines patterns seen #1 and #3 from standard autoencoder.

Finally, we compare the spatial structure of our cluster with known climate patterns from KGC [26]: we observe that #1 overlaps with the humid subtropical climate zone (Cfa) and humid continental climate zone (Dfa). #5 is spatially associated with their semi-arid (BS) and desert (BW) climate types, suggesting that our clusters capture physically meaningful spatial patterns that identify similar climate areas tightly.

Overall, we conclude that the proposed clustering autoencoder generates physically meaningful and spatially comparable cluster patterns, and dramatically reduces computational resources to those from standard autoencoder.

6.4.6 Cluster patterns for recharge and elevation via standard autoencoder

Our results have demonstrated that clusters based on precipitation and ET are largely unchanged over time, show strong geographical features, and group unique physical regimes on ET-PR space. We extend the analysis to investigate the difference of spatial cluster patterns via standard autoencoders that use recharge, and recharge and elevation data respectively in training. Here, we are motivated to analyze how different input variables alter the resulting climate patterns as our standard autoencoder results in having diverse spatial patterns.

As expected based on prior results, at a high level, Fig. 6.12 displays the common longitudinal structure where #1, a cluster with the highest recharge, only locates at the West coast of CONUS, drier clusters (#4 and #5) are mostly distributed at the Central CONUS, and wetter clusters (#2 and #3) are dominant at the east part of Midwest and South as well as East coast of CONUS.

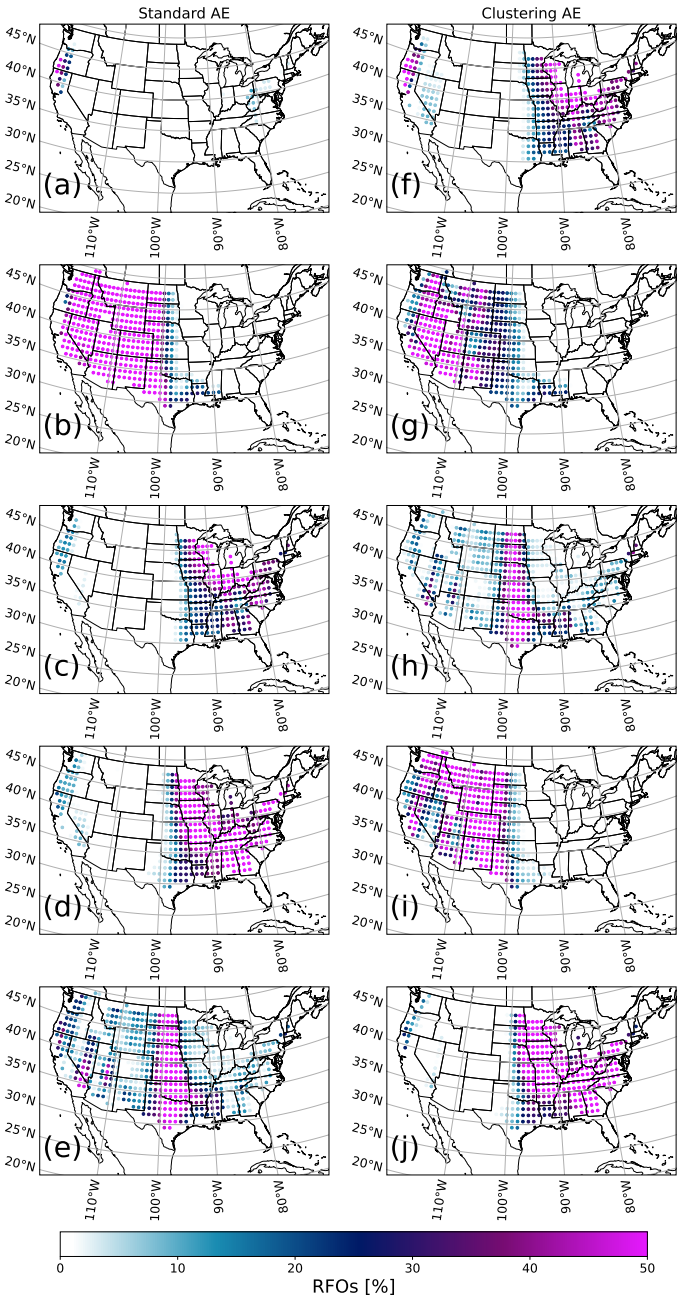
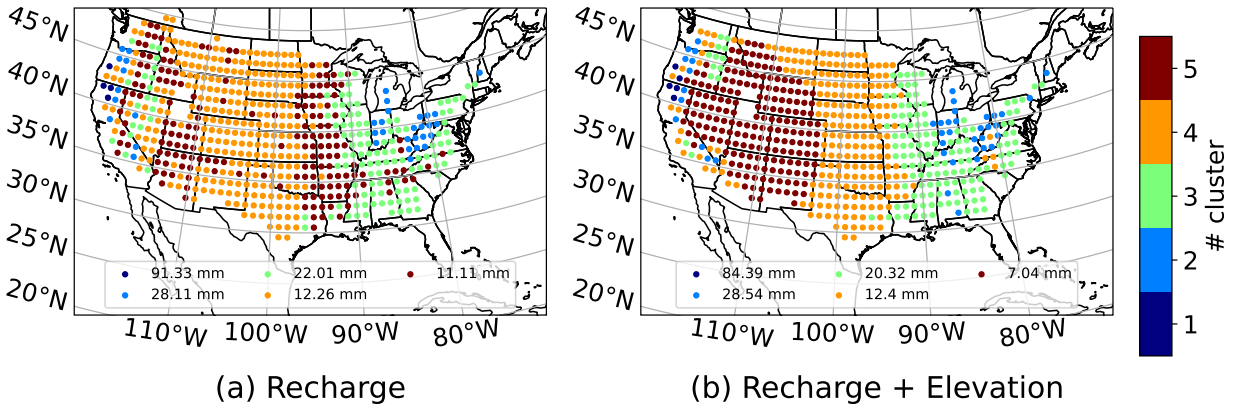


Figure 6.11: In the same way as Fig. 6.9, but plotting the spatial distribution of the relative frequency of occurrence (RFO) for each climate cluster at each patch location from (a)–(e) standard autoencoder and from (f)–(j) clustering autoencoder. Clusters are arranged in rows from top (Cluster 1) to bottom (Cluster 5). RFO of clusters indicates that cluster patterns occur in a specific location instead of distributing evenly. Clusters show strong geographic distinctions, and as shown in Fig. 6.9 those distributions roughly align with known climate patterns.



(a) Recharge (b) Recharge + Elevation

Figure 6.12: Similar to Fig. 6.9, but standard autoencoder and clustering are trained on (a) recharge rate and (b) recharge rate and elevation. Note that, we assign cluster numbers based on within-cluster mean recharge values in descending order. A legend shows the mean recharge rate within-cluster. We observe that training autoencoder with multiple variables helps to produce spatially cohesive cluster patterns, and to make clusters relatively insensitive to outliers from a long-tailed distribution, leading to percent sporadic spatial patterns seen in (a).

In comparison with the precipitation and ET case, cluster patterns resulting solely from recharge values shown in Fig. 6.12 show different spatial structures where two dominant clusters, #3 (22.01 mm/month) and #4 (12.26 mm/month), split the entire CONUS. Because the distribution of patch-mean recharge value is positively skewed and long-tailed as the mode is 3.74 mm/month but the mean is 15.95 mm/month, clusters that group patches around the mode and mean are spatially significant. Here, the resulting clusters tell how different autoencoders capture similar groups of lands depending on input variables over CONUS.

We now evaluate cluster patterns resulting from recharge and elevation (see the right panel (b) in Fig. 6.12). The primary difference of spatial patterns is #5 (i.e., the least perceptible cluster), which clearly captures the topographical effect (see Fig. 6.2) at the Rocky Mountains. Fig. 6.12 also shows that #3 smooths the spatial distribution of clusters at the east part of CONUS (95W–80W) than clusters based on recharge only case distribution, indicating that autoencoder reflects terrain information on the latent representation. That is, adding elevation data in the training data can prevent autoencoder from generating sporadic spatial patterns, which results from grouping outliers in a long-tailed distribution

in recharge rate. The results suggest that the combination of multiple relevant variables may produce more spatially coherent and rich information in the purpose of unsupervised climate classification.

6.4.7 Comparison with conventional classification

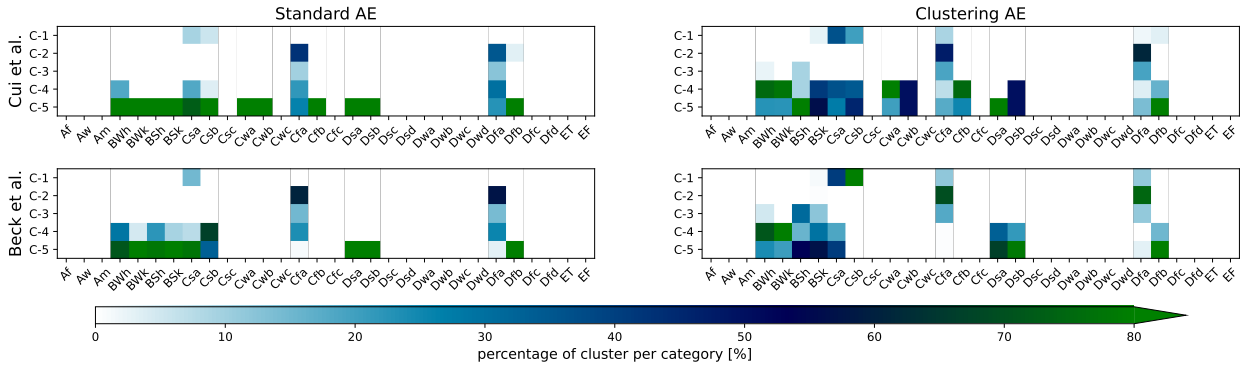


Figure 6.13: Comparison of our two clustering results (column) with two updated versions of 30 KGC subcategories (row) [13, 26]. The color scheme shows the percentage of one of five clusters overlapping on each KGC sub-category. The plots suggest that our approaches group unique climate zones regardless of conventional B, C, and D zones: Cluster 2 is composed of Cfa and Dfa; B, Cs, and Ds are grouped by Cluster 4 and 5; high precipitation area at Csa and Csb is assigned to Cluster 1.

In this section, we conduct a quantitative test to identify the similarities and unique patterns that the autoencoders capture with KG climate classes. We examine the percentage of overlaps between two versions of future 30 KGC climate sub-categories during the late century [26, 13]. We define the percent of cluster per category as the sum of percent from our Cluster 1 to 5 is 100% at each sub-category. We note that since our patches overlap every 1 degree, we aggregate the most frequent KGC class every 1 degree if the dataset provides high-resolution ($<$ 1 degree) data. The results from comparing the future RCP 8.5 projections of our analysis and analyses from two studies demonstrate that our five clusters from the clustering autoencoder have a diversity of cluster arrangements, whereas those from the standard autoencoder put most of the B (Dry), C (Temperate), and D (Continental) subcategories into Cluster 5.

Firstly, when considering the distribution of the KG sub-categories between clusters 1 to 5, there is less distinctiveness within Cs sub-categories. Specifically, based on the clustering autoencoder, for Csa, 36.4% belongs to Cluster 1, 36.4% to Cluster 4, and 27.3% to Cluster 5 from [26] dataset (hereafter **C21**) and 40.7% belongs to Cluster 1, 18.5% to Cluster 4, and 40.7% to Cluster 5 from [13] dataset (hereafter **B18**). However, dissimilarities in Csb, which describes the region with cooler summer compared to Csa, arise from variations in the dataset and different criteria used to define sub-categories in those two studies. For clustering autoencoder based on B18 is predicted to be 100% in Cluster 1, whereas results based on C21 are composed of 20% of Cluster 1, 34.5% of Cluster 4, and 45.5% of Cluster 5. The analysis also reveals that Cluster 1 predominantly overlaps with Csa and Csb sub-categories from C21 and B18. Both sub-categories are characterized by the same precipitation patterns throughout the year where more precipitation falls during the winter than during the summer.

Cfa and Dfa show significant similarities in that they spred over all clusters but the highest percentage is dominant by Cluster 2. Cfa ranges from 42.5 to 69.4% and Dfa ranges from 35.1% to 74.6% in Cluster 2. The similarity accounts for the almost identical definitions of Cfa and Dfa in terms of precipitation. Cluster 2 mainly overlaps with the Cfa and Dfa from both, C21 and B18 datasets, and those are characterized by similar precipitation distribution with no dry seasons which is in agreement with our analysis where Cluster 2 is associated as the cluster with the second wet conditions across the CONUS.

Furthermore, in analyzing how the area of Cluster 3 overlaps with sub-categories from C21 and B18, we may notice that there are lower values of Cfa and Dfa compared to Cluster 2 including additional Bsh, indicating a dryer climate.

Finally, despite the fact that Cfa and Dfa have common cluster percentages, we observe that Dfb exhibits more similarities with Csa, BSh, and Dsa. Figure 6.2 indicates that Dfb distributes on less precipitation and evapotranspiration areas across CONUS, indicating that dry-like sub-categories are scattered among Cluster 4 and 5. More sub-categories with the arid BW (desert) and BS (steppe) overlap with Cluster 4, such as BWh, BWk, Cwa, and

Cfb, while the area of Cluster 5 mainly overlaps with BSh, BSk, Dsa, Dsb, and Dfb from C21 and B18 analyses. This shows us that we successfully identified the arid area across the CONUS as Cluster 4 and 5 are defined as the clusters with the driest conditions. Overall, these findings highlight the complexity of climate clusters generated by autoencoders against rule-based KGC categories.

6.5 Summary

We have presented our unsupervised climate classification approach that reduces the dimensionality of the vast climate simulation data to capture five unique climate patterns via autoencoder techniques, and then provides lightweight climate pattern projections across the continental United States. Our two autoencoders, standard and clustering autoencoder, generate physically reasonable, homogeneous within-cluster, and distinct inter-cluster latent representations. We verify that our clusters show spatially stable climatological patterns along with future extreme precipitation events. The clusters also yield spatial climate patterns, some of which match known climate classes defined by human experts. Our results support the exploratory power of autoencoders and the benefits of extracting only relevant vast amounts of climate simulations compressing by a factor of 660 000.

Our five AI-generated climate clusters are based on a purely data-driven approach without reliance on location, time/seasons, pre-defined variables, and pre-designated thresholds. The method addresses the limitations in complicated classification schema [71, 117] and artificial biases [21] by a comprehensive integration using deep neural networks and data selection. Cluster patterns via autoencoders can reflect the nonlinear combination of different physical features in inputs on the latent space and group distinct data physically and spatially. For example, a standard autoencoder trained on precipitation and ET groups patches into unique physical regimes on ET–PR space, whereas an identical architecture of autoencoder but trained on only recharge values, the difference between precipitation and ET, produces different spatial patterns than the results from precipitation and ET values. The comparison

with KG classification in particular based on results from clustering autoencoder suggests that they identify unique patterns grouping wet and dry patterns across the rule-based conventional sub-categories. We believe that incorporating further climate variables may introduce further complexity of physical information in our AI-generated climate zones.

Our clustering autoencoder shows potential advantages in increasing the capacity of data used for the clustering stage by reducing exponentially increasing clustering time via k–means (e.g., 9.23 times faster to clustering 1 000 000 patches). This leads to improving the generalization of results from data-driven climate classification algorithms by unleashing a larger amount of climate datasets to capture underlying patterns. The preliminary results are able to generate equivalent spatial patterns over CONUS and physical patterns on ET-PR space with clusters from standard autoencoder, whose results are one of the key baselines in unsupervised climate classification. However, we observe that the clustering autoencoder produced imbalanced or equal cluster partitions based on a choice of μ parameter in Eq. 6.3, which adjusts the regularization of cross-entropy. Empirically, the online clustering approach may collapse to the same solution easily [19, 181] based on regularization metrics, hyperparameters, and diverseness of mini-batch. Thus, it will be important to introduce an appropriate evaluation metric for whether clusters produced from clustering autoencoder are physically reasonable. We leave a potential solution in future work.

Overall, we believe that a combination of an autoencoder and clustering approach can help the evaluation of climate patterns immediately without querying large climate datasets. The use of clustering in the latent space of autoencoders on climate data for building representative climate regions can be extended to other domains of applications [171, 186], particularly in climate resilience assessment. We envision applying the unsupervised climate classification workflow under various uncertainty of future climate projections and eventually contributing to demands from stakeholders who need a fast evaluation tool that can handle many possible climate projections.

CHAPTER 7

PIXEL-WISE SELF-ATTENTION SUPER-RESOLUTION NETWORK

In this section, I describe the super-resoluiton neural network that calculates self-attention computation to capture three-dimensional dynamics of weather system. The section is based on the minor modification of workshop and conference publication [77] done in an internship during IBM Research in 2023.

7.1 Related Work

Accurate tracing and monitoring of greenhouse gas (GHG) sources is a key measurement to take action for tackling the mitigation of global warming issues [30]. The higher spatial resolution of wind simulation enables precise tracking of GHG emissions from potential sources, and helps decision-making in various fields such as policymakers, agriculture, and renewable energy. However, high-resolution simulation is not always available due to the demands of computational resources.

Artificial Intelligence (AI), including deep neural networks (DNNs), can reduce the computational cost by upscaling the wind fields from low-resolution (LR) to high-resolution (HR) data. Super-resolution (SR) is one of the solutions to achieve the goal: conventional SR techniques rely on convolutional neural networks (CNNs) to reproduce realistic HR wind fields [151, 184, 78, 83, 190, 2]. However, these CNN-based approaches either perform on 2D data or fall short of capturing the 3D dynamics in weather systems because a convolutional kernel truncates the association between vertical layers, limiting to learning of convection and diurnal cycles induced by incoming solar radiations. Another issue is that widely used weather simulations employ a non-uniform grid in a vertical axis for computational stability. The difference in height between a pair of two adjacent vertical layers becomes larger at higher altitudes. Thus, CNN is not sufficient to capture physics at the same vertical scaling.

To address these issues, we develop a prototype of a novel architecture of physics-informed neural networks that learn multi-scale spatial dynamics. Our goal is to develop a physics-informed neural network that super-resolves 900 m resolution of three-dimensional wind data into 100 m scale high-resolution wind data and captures three-dimensional dynamics in weather systems.

7.2 Methodology

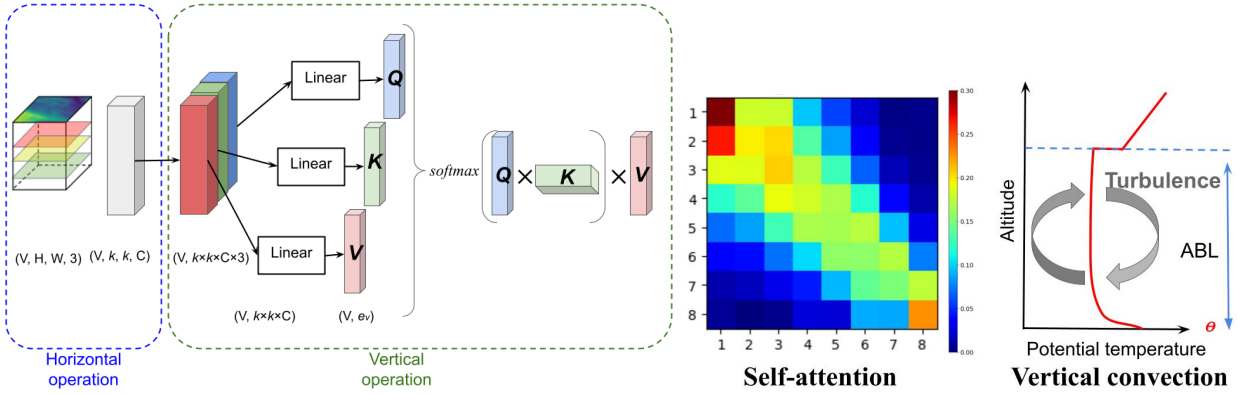
We develop a *pixel-wise self-attention network* (PWA) to learn the three-dimensional dynamics of weather simulations. Using the neural network as a generator, we train a generative network for super-resolving wind velocity fields.

7.2.1 WRF model dataset

The Weather Research & Forecasting Model (WRF) [148] has been widely used for simulating weather systems from mesoscale to turbulence scales. Large-eddy simulation (LES) is often nested into the WRF framework to allow simulating a convective process to capture a more realistic turbulence structure [28]. Outputs from WRF-LES used for this study are nested in 900 m, 300 m, and 100 m horizontal resolution. The outer 900 m and inner 100 m simulations differ in their model physics and our ultimate goal is to super-resolve 900 m LR data to 100 m HR data. This study, however, focuses on synthesizing the 900 m LR data by spatially averaging 100 m HR data as a preliminary step. Our training and testing data are produced by a WRF nested LES at 100 m horizontal scale and 59 vertical layers, generating outputs every five minutes from 00 UTC to 23:59 UTC. The initialization time window is set for two hours starting at 22:00 UTC a day before. We sample training and testing data randomly from one-month simulation outputs from September 2019. We select three wind velocity fields (U, V, W), all at 100 m spatial resolution over 40 km $\times$ 40 km simulation domain, and use only 8 layers from the surface level. We then extract

22 500 training and 2500 testing data, as the smaller geographical images, each of which is 126 pixels × 126 pixels (∼ 12.6 km × 12.6 km) for high-resolution (HR) data. To synthesize low-resolution (LR) data, we average every 9 × 9 pixels, giving 900 m LR data.

7.2.2 Physics-informed pixel-wise self-attention



(a) Diagram of the pixel-wise self-attention module. (b) Physical interpretation.

Figure 7.1: Illustration of the pixel-wise self-attention module: (a) workflow of PWA module. (b) an example 8 × 8 self-attention map (**left**) shows strong (bright colors) and weak (dark colors) signals between adjacent layers, associating convection in weather systems (**right**).

The fluctuation in the atmospheric boundary layer (ABL) height during both day and night significantly impacts the development of vertical convection intensity, which varies across different altitudes. A standard convolutional filter may truncate these signals and be suboptimal to the non-uniform vertical grid system in numerical models for capturing information at the same scale. We introduce a self-attention computation [164] at each grid column to better embed nonlinear association between vertical layers into networks: self-attention inherently computes attention scores for each element in sequences, enabling the representation of signal associations between a given vertical layer and others. This versatility extends beyond the constrains of a convolutional kernel that limits associations solely to neighboring layers above and below.

Figure 7.1 shows the concept of the newly developing pixel-wise self-attention (PWA)

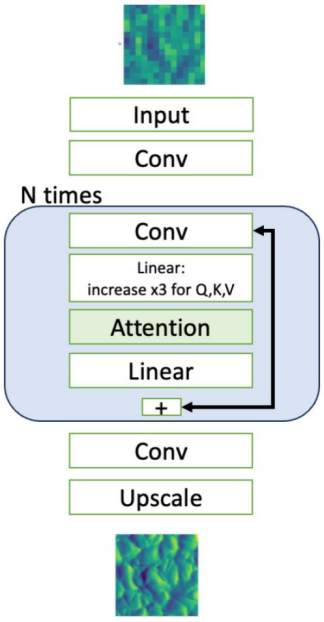


Figure 7.2: Diagram of the architecture of PWA-SR network. We nest the PWA module (highlighted by light blue color) by three times based on our hyperparameter search.

module at a high level. Suppose we have an input image X, a 5D Tensor (N, C, V, H, W) where N, C, V, H, and W depict the size of the mini-batch, the number of variables (i.e., U, V, and W winds), the number of vertical layers, height, and width. PWA module is composed of two parts: *horizontal* and *vertical* operation. For the horizontal operation, we apply the 2D convolutional operation to each image of (V, H, W) by $1 \times 3 \times 3$ convolution kernel. For the vertical operation, we first convert the image shape from (V, H, W, C) to $(V, H \times W \times 3 \cdot C)$ with a linear transformation to create three matrices such that query $Q \in \mathbb{R}^{V \times e_v}$, key $K \in \mathbb{R}^{V \times e_v}$, and value $V \in \mathbb{R}^{V \times e_v}$. We then calculate a self-attention computation by a simple scaled dot product in Equation 7.1 as follows;

$$\text{Attn} = \underbrace{\text{softmax}\left(\frac{Q \cdot K^T}{\sqrt{e_v}}\right)}_{M} \cdot V, \tag{7.1}$$

where e_v is a dimension of intermediate representation and we depict 64 for the dimension. The learned self-attention map M shown in Figure 7.1b shows strong signals in diagonal and near-diagonal elements, indicating that adjacent layers have a stronger effect on each other. In particular, near-surface layers (i.e., left top corner of M) exhibit strong attention in wider elements, and this is associated with daytime convection within the ABL.

Effective learning is achieved through the sparsity of weak signals as well as emphasizing key elements in the self-attention map M depicted in Figure 7.1b. To do so, we add the regularization of M as one of the loss terms as follows:

$$R_i = \sum_{j=1}^{V} M_{ij}^2, \tag{7.2}$$

$$R(M) = \sum_{i=1}^{V} \frac{1}{R_i}, \tag{7.3}$$

where $M_{ij} = \text{softmax}(QK^T/\sqrt{e_v})$ and R depicts a regularization term. The minimization of $R(M)$ in Equation 7.2 indicates the elements of strong attention get larger scores as well as weaker attention reduces the scores. One limitation of this approach is that the range is bounded in $[0, 1]$ at each row. To have more attention to strong signals, and in opposite, suppress the weak ones, we introduce a re-scaling scheme to the self-attention map. Simply, we have a trainable matrix Δ, adjusting the value of self-attention map M by $M^{\text{rescale}} = \Delta \cdot M$. Note that we apply the self-attention map regularization scheme to pre-rescaled M.

7.2.3 PWA SR-GAN: Pixel-wise self-attention super-resolution generative adversarial network

Figure 7.2 illustrates the architecture of our super-resolution network inspired by SR-GAN [80]. Due to the nature of convolutional filters that pass low-frequency modes, the results of neural networks have smoothed structures. To address this, we additionally

include Sobel filter [62] by computing image gradients in horizontal and vertical directions respectively due to the different spatial scales. Overall, we combine multiple loss terms as follows;

$$\mathcal{L} = \mathcal{L}_{\text{content}} + \lambda_R |R(M)| + \underbrace{\lambda_G \sum_{r \in \{u,v\}} |dx/dr - d\hat{x}/dr| + \lambda_{G_v} |dx/dz - d\hat{x}/dz|}_{\text{Image gradient}}, \tag{7.4}$$

where $\mathcal{L}_{\text{content}}$ represents mean squared error $\sum_{x_{LR} \in X_{LR}, x_{HR} \in X_{HR}} (G(x_{LR}) - x_{HR})^2$; $G(\cdot)$ depicts the outputs of our neural network. We have three different weight coefficients: λ_R, λ_G, and λ_{G_v} balances self-attention regularization, horizontal gradient, and vertical gradient terms. We use $(\lambda_R, \lambda_G, \lambda_{G_v}) = (0.01, 50, 100)$ in this work. The training uses Adam optimizer [67] with the learning rate at $1e-3$ on an A-100 GPU for 200 epochs.

CNN-based neural networks are known for filtering low-frequency data in input images and removing high-frequency information [89]. Generative Adversarial Networks (GANs) [41] are our solution to incorporate them into super-resolved images because our training input data x_{LR} has already smoothed out high-frequency information in high-resolution image x_{HR} through averaging the data over 9×9 pixels. Thus, it is challenging to learn the high-frequency data only from low-resolution training inputs. The generator loss in a GAN is typically defined as the negative log-likelihood of the discriminator's output when the generator tries to generate realistic data. The loss term $\mathcal{L}_{\text{adversarial}}$ is often represented as:

$$\mathcal{L}_{\text{adversarial}} = -\mathbb{E}_{\mathbf{x}' \sim p(\mathbf{x}_{\text{LR}})}[\log(D(G(\mathbf{x}')))], \tag{7.5}$$

where $\mathbf{x}'$ is a set of low-resolution wind data sampled from a distribution of low-resolution images; $G(\mathbf{x}')$ is a super-resolved data by the generator; $D(G(\mathbf{x}))$ is a discriminator's output for high-resolution data $\mathbf{x}$; and $p(\mathbf{x}')$ is a prior distribution of $\mathbf{x}'$.

After pre-training PWA-SR network with Equation 7.4, we modify the loss function for

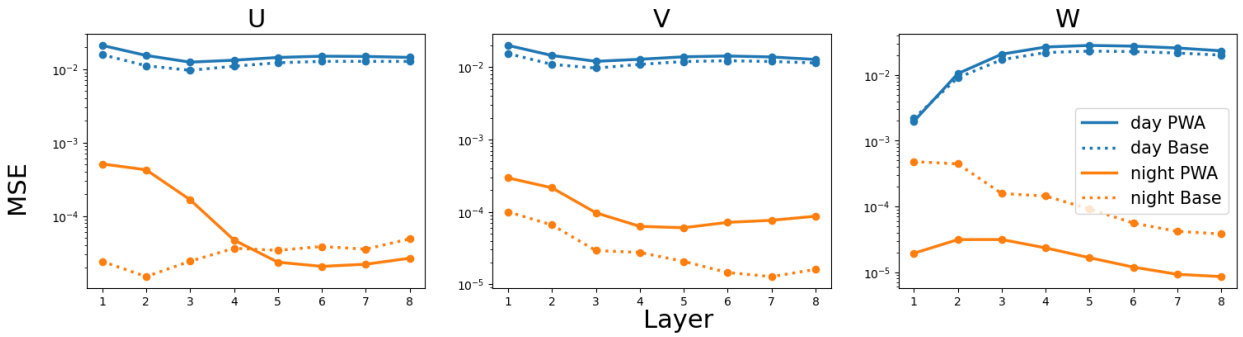


Figure 7.3: Results from mean squared errors (MSE) between days and night time from test dataset. Solid lines represent MSE of pixel-wise attention network and dashed lines represent Residual convolutional network [80].

training a generator by adding an adversarial term into the combined loss function;

$$\mathcal{L} = \mathcal{L}_{\text{content}} + \lambda_G \mathcal{L}_{\text{horizontal}} + \lambda_{G_v} \mathcal{L}_{\text{vertical}} + \lambda_a \mathcal{L}_{\text{adversarial}}. \tag{7.6}$$

For discriminator, we follow the same architecture based on Ledig et al. [80]. See Ledig et al. [80] for further details on the training of GAN.

7.3 Evaluation

We first examine to what extent the pixel-wise attention network captures steady- and unsteady-state over day and night against a standard resnet type network [151]. Figure 7.3 shows the mean square errors from day and night time from PWA and baseline resnet. We see that the mean square errors in day-time have no distinctions, whereas the errors in night-time in particular vertical wind show one to two orders of magnitude smaller than the baseline. Layers closer to the surface are aided by the pixel-wise attention. Horizontal wind does not show the benefit of the architecture designs.

We then investigate whether PWA SR-GAN (hereafter SR-GAN) can generate wind filed data that closely approximates the ground-truth HR image in frequency space. We apply the fast Fourier transform to our 2500 test samples for calculating the power spectrum of images instead of mean squared errors (MSEs) because MSEs do not always evaluate if SR

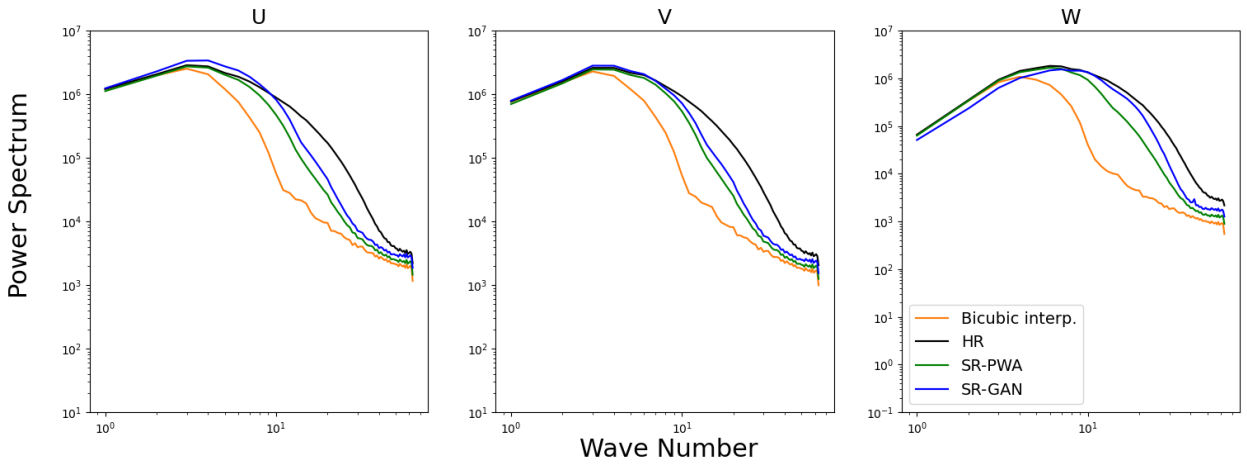


Figure 7.4: Plots of power spectrum analysis for bicubic interpolation, pixel-wise self-attention SR network (SR-PWA), and pixel-wise self-attention SR-GAN (SR-GAN) as a comparison to HR images. SR-GAN outperforms the SR-PWA (green line) for learning high-frequency domains, especially in vertical wind W

models restore high-frequency mode. That is, the test measures how well the model learns the high-frequency mode of wind data through GAN training. Figure 7.4 shows the results of the power spectrum analysis as a function of wave numbers for U, V, and W wind data respectively. The results compare the power spectrum among different approaches: HR data, bicubic interpolation, SR-PWA, and SR-GAN. The fidelity of capturing realistic wind structures improves as their results of the power spectrum align more closely with HR data (black line). Note that a conventional residual CNN yields a result similar to that of SR-PWA (not reported in this paper). We see that the SR-GAN approach (blue line) significantly outperforms the SR-PWA (green line) for learning high-frequency domains, especially in vertical wind W. However, there is a performance trade-off in low to medium-wave numbers when the learning high-frequency domain achieves well.

As part of a qualitative evaluation, we present a snapshot example of wind data from the first layer in Figure 7.5, illustrating the fidelity of super-resolved wind images as a comparison between LR data and HR data. SR-PWA restores major wind structures and the velocity intensities. SR-GAN excels in generating finer wind structures, as evidenced by the results depicted in Figure 7.4.

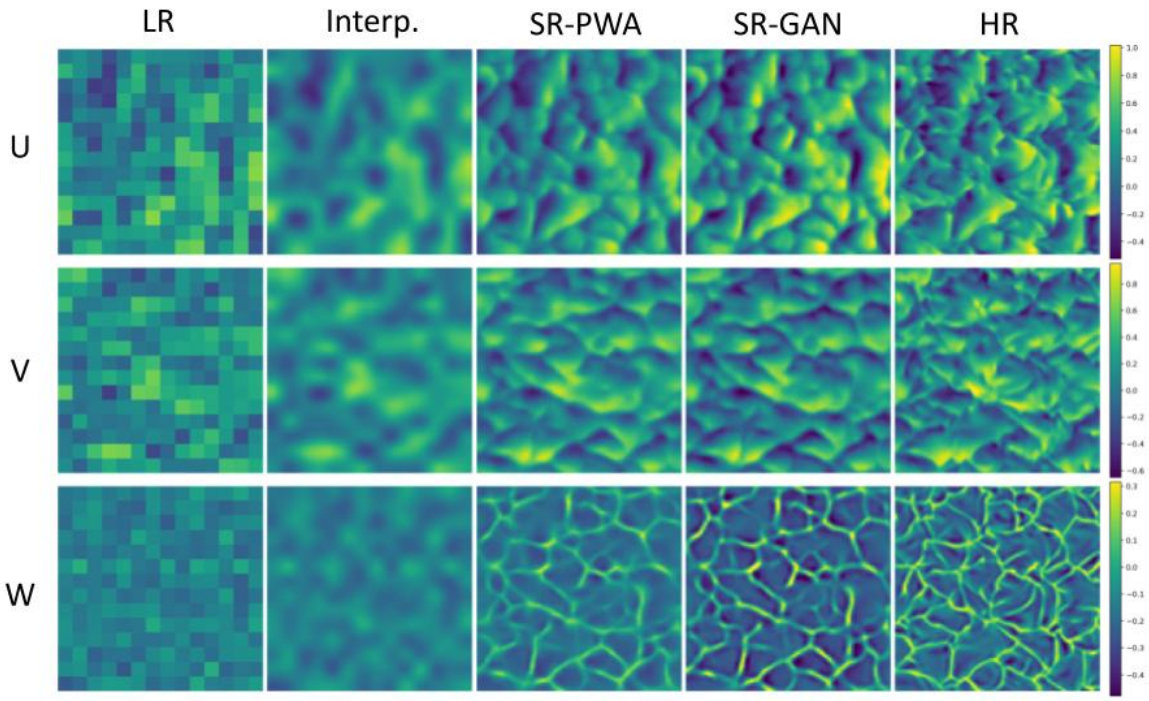


Figure 7.5: Comparison of qualitative test results based on an example snapshot at the first layer. Each row shows the super-resolved or raw images from U, V, or W wind components.

Additionally, we observe that our SR-GAN decreases computational costs ($\sim$ CPU time $\times$ number of cores) by $\times$89.7 times compared with the original WRF-LES. Overall, the results indicate that PWA SR-GAN generates realistic 3D high-dimensional wind fields by lower computing power and resources, enabling simulating advection and diffusion of GHG gas solely using low-resolution WRF simulation outputs.

7.4 Physical consistency

The previous section provides the performance of super-resolution and the minimal proof of to which extent the pixel-wise self-attention network improves the performance for vertical dynamics. Despite regular SR tasks in computer vision applications, the application for scientific data in fluid dynamics also needs to take into consideration whether the resulting SR values suffice the physical consistency. This section discusses the literature and the methodology to validate physical consistency for this target application.

A common criticism of 'physics-informed' super-resolution deep learning attempts is the frequent violation of conservation laws by SR models, despite their visually compelling output. To ensure the physical plausibility of these models, some straightforward approaches involve integrating flow governing equations [38, 40], such as mass conservation laws and fluid divergence/curl (i.e., wind direction), into SR networks through the calculation of conservation laws from high-resolution (HR) results [193, 194]. Alternatively, simpler yet effective methods evaluate whether LR images and spatially averaged HR results adhere to the same physics [39]. The approach can be achieved by incorporating a 'constraint layer' to minimize the mismatch between LR and predicted HR values based on the target physical equations [47]. It is noteworthy that numerical weather and climate models often use different governing equations and the hypothesis of physics based on the spatial resolution of applications [28], raising a caution to apply the matching approach. Furthermore, the design of loss functions [142] is important in ensuring that predicted values align with divergence and curl terms, particularly in wind and fluid dynamics systems where the direction of flow and wind advection are critical.

The ultimate goal in this application aims to super-resolve 900m×900m grid resolution of LR mages to 100m×100m HR images, where WRF regional weather simulations use different dynamical cores between 900m and 100m. Given the constrain, the calculation of mass conservation within resulting HR images is the most straightforward approach.

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{7.7}$$

where ρ depicts the air density; t is a timestep; $\mathbf{u}$ represents three-dimensional velocity vector (u, v, w).

However, calculating governing equations may present technical challenges, particularly when weather simulations utilize non-uniform vertical coordinates. Discretization becomes challenging in such scenarios, as the volume of mathematics involved is non-isotropic. To ease

the condition as the order of magnitude of vertical winds is smaller than horizontal winds, the equation 7.7 may reduce the dimension to horizontal space. In the two-dimensional case, divergence and curl terms are also simplified so that these additional factors could ensure the SR model follows the theory rigorously. Given these difficulties, the introduction of a constraint layer [47] emerges as the most expedient yet effective approach to enhance physical reasonability.

I expect that the physics-constrain may not influence the major wind structure since PWA-SR networks sufficiently create the fields as shown in Figure 7.5, while it can adjust values such as velocity vector from U and V produce compelling results as the ground-truth HR images. One way to investigate whether the resulting wind vector field does not violate physical consistency is to calculate the mismatch of wind velocity vectors. The residual (i.e., differences between predicted values and ground truth) of Wind speeds and streamlines will have sufficiently small errors and random streamline structures.

7.5 Summary

In summary, we show our preliminary investigation of the super-resolved 3D wind structures based on a newly developing SR network that utilizes a self-attention network and a generative model. It enables to incorporating of 3D dynamics of weather systems that are essential to reconstructing physically representative 3D wind fields, and then achieve to generate high-fidelity outputs. This work is a preliminary step toward building a tracing methodology that is capable of simulating the trajectory of GHGs combined with limited observation and reduces the computational overhead. Further algorithmic advancements should improve the accuracy of the methodology.

CHAPTER 8

CONCLUSION

Since AlexNet [73] made a first groundbreaking advancement in deep learning, the broad of AI technologies have been intensively studied to improve both performance of deep learning techniques and contribute to the acceleration of scientific discoveries in a wide variety of science fields. In this thesis, I have studied algorithms and applications of deep neural networks for making a vast volume of climate science datasets more accessible to scientists without the allocation of large HPC resources. I outline my contributions and findings from deep learning-powered techniques as follows.

In Chapter 3, I first introduce an autoencoder with a rotation-invariant loss function approach, instead of optimizing mean squared error to train a common convolutional neural network, to cloud classification. Before the study, the mainstream AI-based cloud classification approaches relied on the supervised learning method, where models are trained with labels based on either threshold-based (i.e., ISCCP: nine types of cloud classes delineated by low/medium/high values for both cloud top pressure and cloud optical thickness) or subjective example-based approaches (WMO: 10 main classes and $>$ 28 sub-classes based on cloud height, texture, shape, and other environmental factors). However, clouds in the real atmosphere are not simplistic and the supervised learning-based cloud classification could oversimple some unknown types of clouds or trends of clouds by classifying them into nine or ten types of categories. The prototype of a self-supervised approach presents challenges in downstream tasks, manifesting in the grouping of clusters based on the orientation of cloud objects, difficulty in discerning their scientific utility, and occasional mismatches with known cloud types. In response to these challenges, I introduce the Rotationally Invariant Cloud Clustering (RICC) algorithm. This algorithm achieves rotationally invariant clustering results, effectively categorizing MODIS cloud images into homogeneous AI-driven cloud classes. The evaluation of RICC is conducted based on five criteria, providing both quantitative and qualitative insights. These criteria serve as effective measures to distinguish

between useful and non-useful autoencoder approaches, contributing to the exploration of self-supervised learning applications in cloud classification.

In Chapter 4, I create the AI-driven Cloud Classification Atlas (AICCA) to provide a compact form of complex cloud information translated into 42 types of clouds, all captured by the RICC algorithm. I then apply the RICC to the 23 years of 872TB of MODIS cloud products or 331 million of roughly 100 km $\times$ 100 km patches (128 $\times$ 128 pixels) - into 42 AI-generated cloud class labels for the investigation of cloud trends, which may not be unknown in previous studies. I have observed that the AICCA classes provide meaningful distinctions by leveraging information on spatial structure. While AICCA cloud classes exhibit consistent physical properties, they cannot be solely reproduced from the mean cloud properties in each patch. Additionally, AICCA classes demonstrate robust geographic and temporal distributions, capturing phenomena such as stratocumulus decks along the West coast of North and South America or high-latitude clouds that appear only in local summer. Case studies indicate that AICCA classes contribute to diagnosing the physical mechanisms driving the evolution of cloud textures. For instance, the percentage of stratocumulus-related clouds associated with open and closed textures changes in response to the Pacific Decadal Oscillation index. I also observe that the change in frequency of occurrences among low-clouds AICCA classes not only matches a recent finding of decreasing cloudiness in a critical part of the subtropical stratocumulus deck over California, but revels the increasing sparse texture of marine stratocumulus clouds. These findings suggest that AICCA can give insight to study trends in cloud classes.

In Chapter 5, I first describe the development of a satellite emulator designed to translate high-resolution simulations into MODIS radiances. The emulator serves as an emulator for satellite simulators. Traditional satellite simulators rely on numerous hypotheses regarding microcloud physics, including assumptions about shapes, reflection properties, and distribution. In contrast, the emulator eliminates these assumptions during computation and establishes a direct mapping between atmospheric variables and radiances. By evaluating SCuBA

workflow, I show that IFS generates moderately greater amount of low clouds compared to satellite observation. I noted that during training, RICC places significant emphasis on texture features, whereas climate models need to accurately replicate finer textures. This comparative analysis sheds light on the nuanced differences in texture representation between the two approaches. As a future research direction, it is required to improve the capability of emulator to decrease the blurriness in their outputs, making the SCuBA be more powerful analytical framework.

In Chapter 6, I develop self-supervised climate classification approach that reduces the dimensionality of the vast climate simulation data to capture five unique climate patterns via autoencoder techniques, and then provides lightweight climate pattern projections across the continental United States. As AICCA provides the gist of cloud information in 42 types as well as patch-average cloud parameters, a combination of an autoencoder and clustering approach can help the evaluation of climate patterns simulated in climate simulation outputs immediately without querying large climate datasets every time. Two autoencoders, standard and clustering autoencoders, generate physically reasonable, homogeneous within-cluster, and distinct inter-cluster latent representations. We verify that our clusters show spatially stable climatological patterns along with future extreme precipitation events. I observed that the clusters also yield spatial climate patterns, some of which match known climate classes defined by human experts. Our results support the exploratory power of autoencoders and the benefits of extracting only relevant vast amounts of climate simulations compressing by a factor of 660 000.

Finally, in Chapter 7, I explore whether a 3D super-resolution neural network to address challenges associated with the computational complexity of resolving high-resolution wind fields. The objective is to access detailed data without the need for extensive computational resources. Traditional super-resolution techniques applied in computer vision are suboptimal for high-resolution weather simulations due to the dynamic 3D nature of weather systems, particularly the variations occurring day and night. Standard convolutional kernels may

not adequately capture vertical motion, and the additional complexity arises from the non-uniform vertical coordinate of wind components needed for computational stability. To overcome these challenges, I introduce an adversarial deep neural network that integrates convolutional neural networks with a spatial self-attention computation module. This innovative approach learns the association of wind velocity with adjacent vertical layers using self-attention computation, rather than relying on 3D convolution filters. The results indicate that the super-resolution model effectively reduces reconstruction errors across vertical layers associated with non-uniform vertical grids, leading to more realistic wind velocity field reproductions compared to standard CNNs.

AI4Science is a promising scientific field to bridge deep neural networks and natural sciences, greatly enforced by the significant improvements in computing hardware. In this thesis, I demonstrate various types of AI4Science applications mainly using self-supervised neural networks to accomplish extraction of useful and meaningful information from climate datasets, which need expensive computing resources without these deep learning-powered techniques. As trillion parameter models are emerging in natural language processing, I envision that large self-supervised learning models, or *Foundation Model* should contribute to capture more complex physics and unknown feedback in climate science. In the future direction, I'm interested in encapsulating Foundation Models into AI4Science applications to tackle the discovery of new science knowledge.

References

[1] Hendrik Andersen, Jan Cermak, Lukas Zipfel, and Timothy A. Myers. Attribution of observed recent decrease in low clouds over the northeastern Pacific to cloud-controlling factors. *Geophysical Research Letters*, 49(3), February 2022. ISSN 0094-8276, 1944-8007. doi: 10.1029/2021GL096498. URL https://onlinelibrary.wiley.com/doi/10.1029/2021GL096498.

[2] Nicolaas J Annau, Alex J Cannon, and Adam H Monahan. Algorithmic hallucinations of near-surface winds: Statistical downscaling with generative adversarial networks to convection-permitting scales. *Artificial Intelligence for the Earth Systems*, 2023. doi: 10.1175/AIES-D-23-0015.1.

[3] R Aparna and Sumam Mary Idicula. Spatio-temporal data clustering using deep learning: A review. In *2022 IEEE International Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, pages 1–10, 2022. doi: 10.1109/EAIS51927.2022.9787701.

[4] D. Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Symposium on Discrete Algorithms*, 2007. doi: 10.5555/1283383.1283494.

[5] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. Technical report, Stanford, 2006.

[6] Sara Atito, Muhammad Awais, and Josef Kittler. Sit: Self-supervised vision transformer. *arXiv preprint arXiv:2104.03602*, 2021.

[7] Yadu Babuji, Anna Woodard, Zhuozhao Li, Daniel S Katz, Ben Clifford, Rohan Kumar, Lukasz Lacinski, Ryan Chard, Justin M Wozniak, Ian Foster, et al. Parsl: Pervasive parallel programming in Python. In *28th International Symposium on High-Performance Parallel and Distributed Computing*, pages 25–36. IEEE, 2019. doi: 10.1145/3307681.3325400.

[8] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt. Sparse shift-invariant representation of local 2D patterns and sequence learning for human action recognition. *21st International Conference on Pattern Recognition*, pages 3823–3826, 2012.

[9] Margot Bador, Julien Boé, Laurent Terray, Lisa V. Alexander, Alexander Baker, Alessio Bellucci, Rein Haarsma, Torben Koenigk, Marie-Pierre Moine, Katja Lohmann, Dian A. Putrasahan, Chris Roberts, Malcolm Roberts, Enrico Scoccimarro, Reinhard Schiemann, Jon Seddon, Retish Senan, Sophie Valcke, and Benoit Vanniere. Impact of higher spatial atmospheric resolution on precipitation extremes over land in global climate models. *Journal of Geophysical Research: Atmospheres*, 125(13), 2020. doi: https://doi.org/10.1029/2019JD032184. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2019JD032184.

[10] Bahman Bahmani, Benjamin Moseley, Andrea Vattani, Ravi Kumar, and Sergei Vassilvitskii. Scalable k-means++. *arXiv preprint arXiv:1203.6402*, 2012.

[11] Richard L Bankert. Cloud classification of AVHRR imagery in maritime regions using a probabilistic neural network. *Journal of Applied Meteorology*, 33(8):909–918, 1994. doi: 10.1175/1520-0450(1994)033⟨0909:CCOAII⟩2.0.CO;2.

[12] Bryan A. Baum, Paul W. Menzel, Richard A.Frey, David C. Tobin, Robert E. Holz, Steve A. Ackerman, Andrew K. Heidinger, and Ping Yang. MODIS cloud-top property refinements for Collection 6. *Journal of Applied Meteorology and Climatology*, 51(6): 1145–1163, 2012. doi: 10.1175/JAMC-D-11-0203.1.

[13] Hylke E Beck, Niklaus E Zimmermann, Tim R McVicar, Noemi Vergopolan, Alexis Berg, and Eric F Wood. Present and future köppen-geiger climate classification maps at 1-km resolution. *Scientific data*, 5(1):1–12, 2018. doi: 10.1038/sdata.2018.214.

[14] Gregory Benton, Marc Finzi, Pavel Izmailov, and Andrew G Wilson. Learning invariances in neural networks from training data. *Advances in neural information processing systems*, 33:17605–17616, 2020.

[15] Purnima Bholowalia and Arvind Kumar. Ebk-means: A clustering technique based on elbow method and k-means in wsn. *International Journal of Computer Applications*, 105(9):17–24, November 2014. doi: 10.5120/18405-9674.

[16] Diego Bueso, Maria Piles, and Gustau Camps-Valls. Nonlinear pca for spatio-temporal analysis of earth observation data. *IEEE Transactions on Geoscience and Remote Sensing*, 58(8):5752–5763, 2020. doi: 10.1109/TGRS.2020.2969813.

[17] Gustau Camps-Valls, Devis Tuia, Xiao Xiang Zhu, and Markus Reichstein. *Deep learning for the Earth Sciences: A comprehensive approach to remote sensing, climate science and geosciences*. John Wiley & Sons, 2021. doi: 10.1002/9781119646181.

[18] M. Caron, P. Bojanowski, Armand Joulin, and M. Douze. Deep clustering for unsupervised learning of visual features. In *European Conference on Computer Vision*, 2018.

[19] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33:9912–9924, 2020.

[20] Anadi Chaman and Ivan Dokmanic. Truly shift-invariant convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3773–3783, 2021.

[21] Henry Fletcher Charles, Nobrega-Olivera Malia, A Alegado Rosanna, K. H. Akutagawa Malia, Burkett Maxine, and Giambelluca Thomas. Fourth national climate assessment, volume ii: Impacts, risks, and adaptation in the united states, 2018. URL https://nca2018.globalchange.gov/.

[22] Ashesh Chattopadhyay, Pedram Hassanzadeh, and Saba Pasha. Predicting clustered weather patterns: A test case for applications of convolutional neural networks to

spatio-temporal climate data. *Scientific reports*, 10(1):1–13, 2020. doi: 10.1038/s41598-020-57897-9.

[23] Gong Cheng, Junwei Han, Peicheng Zhou, and Dong Xu. Learning rotation-invariant and fisher discriminative convolutional neural networks for object detection. *IEEE Transactions on Image Processing*, 28(1):265–278, 2019. doi: 10.1109/TIP.2018.2867198.

[24] David Coe, Mathew Barlow, Laurie Agel, Frank Colby, Christopher Skinner, and Jian-Hua Qian. Clustering analysis of autumn weather regimes in the northeast united states. *Journal of Climate*, 34(18):7587 – 7605, 2021. doi: 10.1175/JCLI-D-20-0243.1. URL https://journals.ametsoc.org/view/journals/clim/34/18/JCLI-D-20-0243.1.xml.

[25] Diyang Cui, Shunlin Liang, and Dongdong Wang. Observed and projected changes in global climate zones based on köppen climate classification. *Wiley Interdisciplinary Reviews: Climate Change*, 12(3):e701, 2021. doi: 10.1002/wcc.701.

[26] Diyang Cui, Shunlin Liang, Dongdong Wang, and Zheng Liu. A 1-km global dataset of historical (1979–2017) and future (2020–2100) köppen-geiger climate classification and bioclimatic variables. *Earth System Science Data Discussions*, pages 1–34, 2021. doi: 10.5194/essd-13-5087-2021.

[27] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

[28] Wim C De Rooy, Peter Bechtold, Kristina Fröhlich, Cathy Hohenegger, Harm Jonker, Dmitrii Mironov, A Pier Siebesma, Joao Teixeira, and Jun-Ichi Yano. Entrainment and detrainment in cumulus convection: An overview. *Quarterly Journal of the Royal Meteorological Society*, 139(670):1–19, 2013. doi: 10.1002/qj.1959.

[29] Porntip Dechpichai, Nuttawadee Jinapang, Pariyakorn Yamphli, Sakulrat Polamnuay, Sittisak Injan, and Usa Humphries. Multivariable panel data cluster analysis of meteorological stations in thailand for enso phenomenon. *Mathematical and Computational Applications*, 27(3):37, 2022. doi: 10.3390/mca27030037.

[30] Ruth DeFries, Frédéric Achard, Sandra Brown, Martin Herold, Daniel Murdiyarso, Bernhard Schlamadinger, and Carlos de Souza Jr. Earth observations for estimating greenhouse gas emissions from deforestation in developing countries. *Environmental science & policy*, 10(4):385–394, 2007. doi: 10.1016/j.envsci.2007.01.010.

[31] L. Denby. Discovering the importance of mesoscale cloud organization through unsupervised classification. *Geophysical Research Letters*, 47(1):e2019GL085190, 2020. doi: 10.1029/2019GL085190.

[32] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[33] Sander Dieleman, Kyle W. Willett, and Joni Dambre. Rotation-invariant convolutional neural networks for galaxy morphology prediction. *Monthly Notices of the Royal Astronomical Society*, 450(2):1441–1459, 04 2015. ISSN 0035-8711. doi: 10.1093/mnras/stv632. URL https://doi.org/10.1093/mnras/stv632.

[34] John P Dunne, Jasmin G John, Alistair J Adcroft, Stephen M Griffies, Robert W Hallberg, Elena Shevliakova, Ronald J Stouffer, William Cooke, Krista A Dunne, Matthew J Harrison, et al. Gfdl's esm2 global coupled climate–carbon earth system models. part i: Physical formulation and baseline simulation characteristics. *Journal of climate*, 25(19):6646–6665, 2012. doi: 10.1175/JCLI-D-11-00560.1.

[35] John P Dunne, Jasmin G John, Elena Shevliakova, Ronald J Stouffer, John P Krasting, Sergey L Malyshev, PCD Milly, Lori T Sentman, Alistair J Adcroft, William Cooke, et al. Gfdl's esm2 global coupled climate–carbon earth system models. part ii: carbon system formulation and baseline simulation characteristics. *Journal of Climate*, 26(7): 2247–2267, 2013. doi: 10.1175/JCLI-D-12-00150.1.

[36] C. Farabet, C. Couprie, Laurent Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35:1915–1929, 2013. doi: 10.1109/TPAMI.2012.231.

[37] Auroop R Ganguly, Devashish Kumar, Poulomi Ganguli, Geoffrey Short, and James Klausner. Climate adaptation informatics: water stress on power production. *Computing in Science & Engineering*, 17(6):53–60, 2015. doi: 10.1109/MCSE.2015.106.

[38] Han Gao, Luning Sun, and Jian-Xun Wang. Super-resolution and denoising of fluid flow using physics-informed convolutional neural networks without high-resolution labels. *Physics of Fluids*, 33(7), 2021.

[39] Andrew Geiss and Joseph C Hardin. Strictly enforcing invertibility and conservation in cnn-based super resolution for scientific datasets. *Artificial Intelligence for the Earth Systems*, 2(1):e210012, 2023.

[40] Andrew Geiss, Sam J Silva, and Joseph C Hardin. Downscaling atmospheric chemistry simulations with physically consistent deep learning. *Geoscientific Model Development*, 15(17):6677–6694, 2022.

[41] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. doi: 10.1145/3422622.

[42] J. R. Greaves and D. T. Chang. Technique development to permit optimum use of satellite radiation data. Final report on NASA Contract N62306-69-C-0227, Goddard Space Flight Center, Greenbelt, Maryland, 1970.

[43] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad

Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

[44] Liam Gumley, Jacques Descloitres, and Jeffrey Schmaltz. Creating reprojected true color modis images: A tutorial. *University of Wisconsin–Madison*, 19, 2003.

[45] Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. Synthetic data for text localisation in natural images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2315–2324, 2016.

[46] Carole J Hahn, William B Rossow, and Stephen G Warren. ISCCP cloud properties associated with standard cloud types identified in individual surface observations. *Journal of Climate*, 14(1):11–28, 2001. doi: 10.1175/1520-0442(2001)014⟨0011:ICPAWS⟩2.0.CO;2.

[47] Paula Harder, Qidong Yang, Venkatesh Ramesh, Prasanna Sattigeri, Alex Hernandez-Garcia, Campbell Watson, Daniela Szwarcman, and David Rolnick. Generating physically-consistent high-resolution climate data with hard-constrained neural networks. *arXiv preprint arXiv:2208.05424*, 2022.

[48] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

[49] Hans Hersbach, Bill Bell, Paul Berrisford, Shoji Hirahara, András Horányi, Joaquín Muñoz-Sabater, Julien Nicolas, Carole Peubey, Raluca Radu, Dinand Schepers, Adrian Simmons, Cornel Soci, Saleh Abdalla, Xavier Abellan, Gianpaolo Balsamo, Peter Bechtold, Gionata Biavati, Jean Bidlot, Massimo Bonavita, Giovanna Chiara, Per Dahlgren, Dick Dee, Michail Diamantakis, Rossana Dragani, Johannes Flemming, Richard Forbes, Manuel Fuentes, Alan Geer, Leo Haimberger, Sean Healy, Robin J. Hogan, Elías Hólm, Marta Janisková, Sarah Keeley, Patrick Laloyaux, Philippe Lopez, Cristina Lupu, Gabor Radnoti, Patricia Rosnay, Iryna Rozum, Freja Vamborg, Sebastien Villaume, and Jean-Noël Thépaut. The ERA5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, 146(730):1999–2049, July 2020. ISSN 0035-9009, 1477-870X. doi: 10.1002/qj.3803. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/qj.3803.

[50] Dennis E Hinkle, William Wiersma, and Stephen G Jurs. *Applied Statistics for the Behavioral Sciences*. Houghton Mifflin, 2003. doi: 10.2307/1164825.

[51] Geoffrey E Hinton and S Zemel Richard. Autoencoders, minimum description length and Helmholtz free energy. In *Advances in Neural Information Processing Systems 6*, pages 3–10. Morgan-Kaufmann, 1994.

[52] Geoffrey E. Hinton, A. Krizhevsky, and S. Wang. Transforming auto-encoders. In *International Conference on Artificial Neural Networks*, pages 44–51. Springer, 2011.

[53] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985. doi: 10.1007/BF01908075.

[54] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *32nd International Conference on Machine Learning*, pages 448–456, 2015. doi: 10.48550/arXiv.1502.03167.

[55] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[56] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *28th International Conference on Neural Information Processing Systems*, NIPS'15, page 2017–2025, 2015.

[57] Pallavi Jain, Bianca Schoen-Phelan, and Robert Ross. Self-supervised learning for invariant representations from multi-spectral and sar images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 15:7797–7808, 2022. doi: 10.1109/JSTARS.2022.3204888.

[58] Christian Jakob and George Tselioudis. Objective identification of cloud regimes in the tropical western pacific. *Geophysical Research Letters*, 30(21), 2003. doi: 10.1029/2003GL018367. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2003GL018367.

[59] Ylva Jansson and Tony Lindeberg. Scale-invariant scale-channel networks: Deep networks that generalise to previously unseen scales. *Journal of Mathematical Imaging and Vision*, 64(5):506–536, 2022. doi: 10.1007/s10851-022-01082-2.

[60] Quanjun Jiao, Qi Sun, Bing Zhang, Wenjiang Huang, Huichun Ye, Zhaoming Zhang, Xiao Zhang, and Binxiang Qian. A random forest algorithm for retrieving canopy chlorophyll content of wheat and soybean trained with prosail simulations using adjusted average leaf angle. *Remote Sensing*, 14(1), 2022. ISSN 2072-4292. doi: 10.3390/rs14010098. URL https://www.mdpi.com/2072-4292/14/1/98.

[61] Chen Jin, Ruoqian Liu, Zhengzhang Chen, William Hendrix, Ankit Agrawal, and Alok Choudhary. A scalable hierarchical clustering algorithm using Spark. In *IEEE First International Conference on Big Data Computing Service and Applications*, pages 418–426. IEEE, 2015. doi: 10.1109/BigDataService.2015.67.

[62] Zhang Jin-Yu, Chen Yan, and Huang Xian-Xiang. Edge detection of images based on improved sobel operator and genetic algorithms. In *2009 International Conference on Image Analysis and Signal Processing*, pages 31–35, 2009. doi: 10.1109/IASP.2009.5054605.

[63] S. C. Johnson. Hierarchical clustering schemes. *Psychometrika*, 32:241–254, 1967. doi: 10.1007/BF02289588.

[64] Christopher O Justice, Eric Vermote, John RG Townshend, Ruth Defries, David P Roy, Dorothy K Hall, Vincent V Salomonson, Jeffrey L Privette, George Riggs, Alan Strahler, W. Lucht, R.B. Myneni, Y. Knyazikhin, S.W. Running, R.R. Nemani, Zhengming Wan, A.R. Huete, W. van Leeuwen, R.E. Wolfe, L. Giglio, J. Muller, P. Lewis, and M.J. Barnsley. The Moderate Resolution Imaging Spectroradiometer (MODIS): Land remote sensing for global change research. *IEEE Transactions on Geoscience and Remote Sensing*, 36(4):1228–1249, 1998. doi: 10.1109/36.701075.

[65] Angjoo Kanazawa, Abhishek Sharma, and David W. Jacobs. Locally scale-invariant convolutional neural networks. In *Deep Learning and Representation Learning Workshop: Conference on Neural Information Processing Systems*, volume abs/1412.5104, 2014.

[66] Ambarish V Karmalkar, Jeanne M Thibeault, Alexander M Bryan, and Anji Seth. Identifying credible and diverse gcms for regional climate change studies—case study: Northeastern united states. *Climatic Change*, 154(3):367–386, 2019. doi: 10.1007/s10584-019-02411-y.

[67] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[68] Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *2nd International Conference on Learning Representations*, 2014. URL http://arxiv.org/abs/1312.6114.

[69] Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990. doi: 10.1109/5.58325.

[70] W. Köppen and R. Geiger. Das geographische system der klimate. handbuch der klimatologie. 1936.

[71] Wladimir Köppen. Die wärmezonen der erde, nach der dauer der heissen, gemässigten und kalten zeit und nach der wirkung der wärme auf die organische welt betrachtet. *Meteorologische Zeitschrift*, 1(21):5–226, 1884.

[72] Mark A Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal*, 37(2):233–243, 1991. doi: 10.1002/aic.690370209.

[73] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[74] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017. doi: 10.1145/3065386.

[75] Takuya Kurihana, Elisabeth Moyer, Rebecca Willett, Davis Gilton, and Ian Foster. Data-driven cloud clustering via a rotationally invariant autoencoder. *IEEE*

Transactions on Geoscience and Remote Sensing, 60:1–25, 2021. URL https://doi.org/10.1109/TGRS.2021.3098008.

[76] Takuya Kurihana, Elisabeth J. Moyer, and Ian T. Foster. AICCA: AI-driven cloud classification atlas. *Remote Sensing*, 14(22), 2022. ISSN 2072-4292. doi: 10.3390/rs14225690. URL https://www.mdpi.com/2072-4292/14/22/5690.

[77] Takuya Kurihana, Kyongmin Yeo, Daniela Szwarcman, Bruce Elmegreen, Karthik Mukkavilli, Johannes Schmude, and Levente Klein. A 3d super-resolution of wind fields via physics-informed pixel-wise self-attention generative adversarial network. *arXiv preprint arXiv:2312.13212*, 2023.

[78] Rupa Kurinchi-Vendhan, Björn Lütjens, Ritwik Gupta, Lucien Werner, and Dava Newman. Wisosuper: Benchmarking super-resolution methods on wind and solar data. *arXiv preprint arXiv:2109.08770*, 2021.

[79] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

[80] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[81] Jonathan Lee, Ronald C Weger, Sailes K Sengupta, and Ronald M Welch. A neural network approach to cloud classification. *IEEE Transactions on Geoscience and Remote Sensing*, 28(5):846–855, 1990. doi: 10.1109/36.58972.

[82] Arianna Libera, Felipe PJ de Barros, Boris Faybishenko, Carol Eddy-Dilek, Miles Denham, Konstantin Lipnikov, David Moulton, Barbara Maco, and Haruko Wainwright. Climate change impact on residual contaminants under sustainable remediation. *Journal of contaminant hydrology*, 226:103518, 2019. doi: 10.1016/j.jconhyd.2019.103518.

[83] Jia Liu, Yongjian Sun, Kaijun Ren, Yanlai Zhao, Kefeng Deng, and Lizhe Wang. A spatial downscaling approach for windsat satellite sea surface wind based on generative adversarial networks and dual learning scheme. *Remote Sensing*, 14(3):769, 2022. doi: 10.3390/rs14030769.

[84] Yunjie Liu, Evan Racah, Joaquin Correa, Amir Khosrowshahi, David Lavers, Kenneth Kunkel, Michael Wehner, William Collins, et al. Application of deep convolutional neural networks for detecting extreme weather in climate datasets. *arXiv preprint arXiv:1605.01156*, 2016.

[85] Suhas Lohit and Shubhendu Trivedi. Rotation-invariant autoencoders for signals on spheres. *arXiv preprint arXiv:2012.04474*, 2020. doi: 10.48550/arXiv.2012.04474.

[86] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[87] L. V. D. Maaten and Geoffrey E. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008. URL http://jmlr.org/papers/v9/vandermaaten08a.html.

[88] Barbara Maco, Paul Bardos, Frederic Coulon, Emerald Erickson-Mulanax, Lara J Hansen, Melissa Harclerode, Deyi Hou, Eric Mielbrecht, Haruko M Wainwright, Tetsuo Yasutaka, et al. Resilient remediation: Addressing extreme weather and climate change, creating community value. *Remediation Journal*, 29(1):7–18, 2018. doi: 10.1002/rem.21585.

[89] Salma Abdel Magid, Yulun Zhang, Donglai Wei, Won-Dong Jang, Zudi Lin, Yun Fu, and Hanspeter Pfister. Dynamic high-pass filtering and multi-spectral attention for image super-resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4288–4297, 2021.

[90] Willem J Marais, Robert E Holz, Jeffrey S Reid, and Rebecca M Willett. Leveraging spatial textures, through machine learning, to identify aerosols and distinct cloud types from multispectral observations. *Atmospheric Measurement Techniques*, 13(10):5459–5480, 2020. doi: 10.5194/amt-13-5459-2020.

[91] Jonathan Masci, U. Meier, Dan C. Ciresan, and J. Schmidhuber. Stacked convolutional auto-encoders for hierarchical feature extraction. In *International Conference on Artificial Neural Networks*, 2011. doi: 10.1007/978-3-642-21735-7_7.

[92] Valérie Masson-Delmotte, Panmao Zhai, Anna Pirani, Sarah L Connors, Clotilde Péan, Sophie Berger, Nada Caud, Y Chen, L Goldfarb, MI Gomis, et al. Climate change 2021: the physical science basis. *Contribution of working group I to the sixth assessment report of the intergovernmental panel on climate change*, 2, 2021.

[93] Hirohiko Masunaga, Toshihisa Matsui, Wei-kuo Tao, Arthur Y Hou, Christian D Kummerow, Teruyuki Nakajima, Peter Bauer, William S Olson, Miho Sekiguchi, and Takashi Y Nakajima. Satellite data simulator unit: A multisensor, multispectral satellite simulator package. *Bulletin of the American Meteorological Society*, 91(12):1625–1632, 2010. doi: 10.1175/2010BAMS2809.1.

[94] Tadashi Matsuo and Nobutaka Shimada. Construction of latent descriptor space and inference model of hand-object interactions. *IEICE TRANSACTIONS on Information and Systems*, 100(6):1350–1359, 2017. doi: 10.1587/transinf.2016EDP7410.

[95] Tadashi Matsuo, Hiroya Fukuhara, and Nobutaka Shimada. Transform invariant auto-encoder. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2359–2364, 2017. doi: 10.1109/IROS.2017.8206047.

[96] Edwin P Maurer, Levi Brekke, Tom Pruitt, and Philip B Duffy. Fine-resolution climate projections enhance regional climate change impact studies, 2007.

[97] Adrian J McDonald, John J Cassano, Ben Jolly, Simon Parsons, and Alex Schuddeboom. An automated satellite cloud classification scheme using self-organizing maps: Alternative isccp weather states. *Journal of Geophysical Research: Atmospheres*, 121(21):13–009, 2016. doi: https://doi.org/10.1002/2016JD025199.

[98] AJ McDonald and Simon Parsons. A comparison of cloud classification methodologies: Differences between cloud and dynamical regimes. *Journal of Geophysical Research: Atmospheres*, 123(19):11–173, 2018. doi: https://doi.org/10.1029/2018JD028595.

[99] Marc J Metzger, Robert GH Bunce, Rob HG Jongman, Roger Sayre, Antonio Trabucco, and Robert Zomer. A high-resolution bioclimate map of the world: a unifying framework for global biodiversity research and monitoring. *Global Ecology and Biogeography*, 22(5):630–638, 2013. doi: 10.1111/geb.12022.

[100] M Mittermeier, M Weigert, D Rügamer, H Küchenhoff, and R Ludwig. A deep learning based classification of atmospheric circulation types over europe: projection of future changes in a cmip6 large ensemble. *Environmental Research Letters*, 17(8):084021, 2022. doi: 10.1088/1748-9326/ac8068.

[101] MODIS Characterization Support Team. Modis/terra 1km calibrated radiances product, 2017.

[102] MODIS Characterization Support Team. MODIS 1km calibrated radiances product. NASA MODIS adaptive processing system. Goddard Space Flight Center, USA, 2017.

[103] MODIS Characterization Support Team. Modis/aqua 1km calibrated radiances product, 2017.

[104] Veronica S Moertini, Gde W Suarjana, Liptia Venica, and Gede Karya. Big data reduction technique using parallel hierarchical agglomerative clustering. *IAENG International Journal of Computer Science*, 45(1), 2018.

[105] Nicholas Monath, Kumar Avinava Dubey, Guru Guruganesh, Manzil Zaheer, Amr Ahmed, Andrew McCallum, Gokhan Mergen, Marc Najork, Mert Terzihan, Bryon Tjanaka, Yuan Wang, and Yuchen Wu. Scalable hierarchical agglomerative clustering. In *27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1245–1255, 2021. doi: 10.1145/3447548.3467404.

[106] Griffin Mooers, Mike Pritchard, Tom Beucler, Prakhar Srivastava, Harshini Mangipudi, Liran Peng, Pierre Gentine, and Stephan Mandt. Comparing storm resolving models and climates via unsupervised machine learning. *Scientific Reports*, 13(1):22365, 2023.

[107] Aaron Muhlbauer, I. L. McCoy, and R. Wood. Climatology of stratocumulus cloud morphologies: Microphysical properties and radiative effects. *Atmospheric Chemistry and Physics*, 14:6695–6716, 2014. doi: 10.5194/acp-14-6695-2014.

[108] V. Nair and Geoffrey E. Hinton. Rectified linear units improve restricted Boltzmann machines. In *International Conference on Machine Learning*, 2010.

[109] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010.

[110] Matthew R Norman, David A Bader, Christopher Eldred, Walter M Hannah, Benjamin R Hillman, Christopher R Jones, Jungmin M Lee, LR Leung, Isaac Lyngaas, Kyle G Pressel, Sarat Sreepathi, Mark A Taylor, and Xingqiu Yuan. Unprecedented cloud resolution in a gpu-enabled full-physics atmospheric climate simulation on olcf's summit supercomputer. *The International Journal of High Performance Computing Applications*, 36(1):93–105, 2022. doi: 10.1177/10943420211027539.

[111] Saeid Ojaghi, Yacine Bouroubi, Samuel Foucher, Martin Bergeron, and Cedric Seynat. Deep learning-based emulation of radiative transfer models for top-of-atmosphere brdf modelling using sentinel-3 olci. *Remote Sensing*, 15(3), 2023. ISSN 2072-4292. doi: 10.3390/rs15030835. URL https://www.mdpi.com/2072-4292/15/3/835.

[112] Baoxiang Pan, Gemma J Anderson, André Goncalves, Donald D Lucas, Céline JW Bonfils, Jiwoo Lee, Yang Tian, and Hsi-Yen Ma. Learning to correct climate projection biases. *Journal of Advances in Modeling Earth Systems*, 13(10):e2021MS002509, 2021. doi: 10.1029/2021MS002509.

[113] The pandas development team. pandas-dev/pandas: Pandas, February 2020. URL https://doi.org/10.5281/zenodo.3509134.

[114] Christina Papagiannopoulou, Diego G Miralles, Matthias Demuzere, Niko EC Verhoest, and Willem Waegeman. Global hydro-climatic biomes identified via multitask learning. *Geoscientific Model Development*, 11(10):4139–4153, 2018. doi: 10.5194/gmd-11-4139-2018.

[115] Deepak Pathak, Philipp Krähenbühl, J. Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2536–2544, 2016.

[116] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[117] Murray C Peel, Brian L Finlayson, and Thomas A McMahon. Updated world map of the köppen-geiger climate classification. *Hydrology and earth system sciences*, 11(5): 1633–1644, 2007. doi: 10.5194/hess-11-1633-2007.

[118] Ruth Petrie, Sébastien Denvil, Sasha Ames, Guillaume Levavasseur, Sandro Fiore, Chris Allen, Fabrizio Antonio, KatharinaPier Berger, Pierre-Antoine Bretonnière, Luca Cinquini, et al. Coordinating an operational data distribution network for cmip6 data. *Geoscientific Model Development Discussions*, 2020:1–22, 2020.

[119] Steven Platnick, Kerry G. Meyer, Michael D. King, Benjamin Marchan, Thomas G. Arnold, Zhibo Zhang, Paul A. Hubanks, Robert E. Holz, Ping Yang, William L.

Ridgway, and Jérôme Riedi. The MODIS cloud optical and microphysical products: Collection 6 updates and examples from Terra and Aqua. *IEEE Transactions on Geoscience and Remote Sensing*, 55(1):502–525, 2017. doi: 10.1109/TGRS.2016.2610522.

[120] PRISM-Climate-Group. Prism gridded climate data, 2014. URL https://prism.oregonstate.edu.

[121] Alexandra Puchko, Robert Link, Brian Hutchinson, Ben Kravitz, and Abigail Snyder. Deepclimgan: A high-resolution climate data generator. *arXiv preprint arXiv:2011.11705*, 2020.

[122] Evan Racah, C. Beckham, Tegan Maharaj, S. Kahou, Prabhat, and C. Pal. ExtremeWeather: A large-scale climate dataset for semi-supervised detection, localization, and understanding of extreme weather events. In *Advances in Neural Information Processing Systems*, 2017.

[123] Evan Racah, Christopher Beckham, Tegan Maharaj, Samira Ebrahimi Kahou, Mr Prabhat, and Chris Pal. Extremeweather: A large-scale climate dataset for semi-supervised detection, localization, and understanding of extreme weather events. *Advances in neural information processing systems*, 30, 2017.

[124] Preesan Rakwatin, Wataru Takeuchi, and Yoshifumi Yasuoka. Stripe noise reduction in MODIS data by combining histogram matching with facet filter. *IEEE Transactions on Geoscience and Remote Sensing*, 45(6):1844–1856, 2007. doi: 10.1109/TGRS.2007.895841.

[125] William M Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850, 1971. doi: 10.1080/01621459.1971.10482356.

[126] S. Rasp, H. Schulz, S. Bony, and B. Stevens. Combining crowd-sourcing and deep learning to understand meso-scale organization of shallow convection, 2019. ArXiv.

[127] SR Reclamation. Downscaled cmip3 and cmip5 climate and hydrology projections: Release of downscaled cmip5 climate projections, comparison with preceding information, and summary of user needs, 2013.

[128] Russ Rew and Glenn Davis. Netcdf: An interface for scientific data access. *IEEE Computer Graphics and Applications*, 10(4):76–82, 1990. doi: 10.1109/38.56302.

[129] George A Riggs, Dorothy K Hall, and Miguel O Román. Modis snow products collection 6 user guide. *National Snow and Ice Data Center: Boulder, CO, USA*, 66, 2015. https://modis-snow-ice.gsfc.nasa.gov/uploads/C6_MODIS_Snow_User_Guide.pdf.

[130] Matthew Rocklin. Dask: Parallel computation with blocked algorithms and task scheduling. In *Proceedings of the 14th python in science conference*, volume 130, page 136. SciPy Austin, TX, 2015. doi: 10.25080/Majora-7b98e3ed-013.

[131] W. B. Rossow and R. A. Schiffer. ISCCP cloud data products. *Bulletin of the American Meteorological Society*, 71:2–20, 1991. doi: 10.1175/1520-0477(1991)072⟨0002:ICDP⟩2.0.CO;2.

[132] William B Rossow and Robert A Schiffer. Advances in understanding clouds from ISCCP. *Bulletin of the American Meteorological Society*, 80(11):2261–2288, 1999. doi: 10.1175/1520-0477(1999)080⟨2261:AIUCFI⟩2.0.CO;2.

[133] William B Rossow, Alison W Walker, and Leonid C Garder. Comparison of isccp and other cloud amounts. *Journal of Climate*, 6(12):2394–2418, 1993. doi: 10.1175/1520-0442(1993)006⟨2394:COIAOC⟩2.0.CO;2.

[134] DA Sachindra and BJC Perera. Statistical downscaling of general circulation model outputs to precipitation accounting for non-stationarities in predictor-predictand relationships. *PloS one*, 11(12):e0168701, 2016.

[135] Mayu Sakurada and Takehisa Yairi. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Machine Learning for Sensory Data Analysis*, pages 4–11. ACM, 2014. ISBN 9781450331593. doi: 10.1145/2689746.2689747.

[136] Kenneth Sassen and Zhien Wang. Classifying clouds around the globe with the CloudSat radar: 1 year of results. *Geophysical Research Letters*, 35(4):L04805, 2008. doi: 10.1029/2007GL032591. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2007GL032591.

[137] R. A. Schiffer and W. B. Rossow. The International Satellite Cloud Climatology Project (ISCCP): The first project of the World Climate Research Programme. *Bulletin of the American Meteorological Society*, 64:779–784, 1983. doi: 10.1175/1520-0477-64.7.779.

[138] Victor Schmidt, Alexandra Sasha Luccioni, Mélisande Teng, Tianyu Zhang, Alexia Reynaud, Sunand Raghupathi, Gautier Cosne, Adrien Juraver, Vahe Vardanyan, Alex Hernandez-Garcia, et al. Climategan: Raising climate change awareness by generating images of floods. *arXiv preprint arXiv:2110.02871*, 2021.

[139] Tapio Schneider, João Teixeira, Christopher S. Bretherton, Florent Brient, Kyle G. Pressel, Christoph Schär, and A. Pier Siebesma. Climate goals and computing the future of clouds. *Nature Climate Change*, 7(1):3–5, January 2017. ISSN 1758-678X, 1758-6798. doi: 10.1038/nclimate3190. URL http://www.nature.com/articles/nclimate3190.

[140] Alex Schuddeboom, Adrian J McDonald, Olaf Morgenstern, Mike Harvey, and Simon Parsons. Regional regime-based evaluation of present-day general circulation model cloud simulations using self-organizing maps. *Journal of Geophysical Research: Atmospheres*, 123(8):4259–4272, 2018. doi: 10.1002/2017JD028196.

[141] Alex Schuddeboom, Vidya Varma, Adrian J McDonald, Olaf Morgenstern, Mike Harvey, Simon Parsons, Paul Field, and Kalli Furtado. Cluster-based evaluation of model compensating errors: A case study of cloud radiative effect in the

southern ocean. *Geophysical Research Letters*, 46(6):3446–3453, 2019. doi: 10.1029/2018GL081686.

[142] Tsuyoshi Thomas Sekiyama, Syugo Hayashi, Ryo Kaneko, and Ken-ichi Fukui. Surrogate downscaling of mesoscale wind fields using ensemble super-resolution convolutional neural networks. *Artificial Intelligence for the Earth Systems*, pages 1–35, 2023.

[143] Alexander Sergeev and Mike Del Balso. Horovod: fast and easy distributed deep learning in tensorflow. *arXiv preprint arXiv:1802.05799*, 2018.

[144] Xu Shen, Xinmei Tian, Anfeng He, Shaoyan Sun, and Dacheng Tao. Transform-invariant convolutional neural networks for image classification and search. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 1345–1354, 2016. doi: 10.1145/2964284.2964316.

[145] C. Shi, Chunheng Wang, Y. Wang, and B. Xiao. Deep convolutional activations-based features for ground-based cloud classification. *IEEE Geoscience and Remote Sensing Letters*, 14:816–820, 2017. doi: 10.1109/LGRS.2017.2681658.

[146] K. Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015. https://arxiv.org/abs/1409.1556.

[147] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[148] William C Skamarock, Joseph B Klemp, Jimy Dudhia, David O Gill, Dale M Barker, Michael G Duda, Xiang-Yu Huang, Wei Wang, Jordan G Powers, et al. A description of the advanced research wrf version 3. *NCAR technical note*, 475:113, 2008. doi: 10.5065/D68S4MVH.

[149] Kihyuk Sohn and H. Lee. Learning invariant representations with local transformations. In *International Conference on Machine Learning*, 2012.

[150] S Solomon, D Qin, M Manning, Z Chen, M Marquis, KB Averyt, M Tignor, and HL Miller. Climate models and their evaluation. *Climate change 2007: The physical science basis*, 2007. URL http://www.ipcc.

[151] Karen Stengel, Andrew Glaws, Dylan Hettinger, and Ryan N King. Adversarial super-resolution of climatological wind and solar data. *Proceedings of the National Academy of Sciences*, 117(29):16805–16815, 2020. doi: 10.1073/pnas.1918964117.

[152] Bjorn Stevens, Masaki Satoh, Ludovic Auger, Joachim Biercamp, Christopher S Bretherton, Xi Chen, Peter Düben, Falko Judt, Marat Khairoutdinov, Daniel Klocke, et al. Dyamond: the dynamics of the atmospheric general circulation modeled on non-hydrostatic domains. *Progress in Earth and Planetary Science*, 6(1):1–17, 2019.

[153] Bjorn Stevens, Sandrine Bony, Hélène Brogniez, Laureline Hentgen, Cathy Hohenegger, Christoph Kiemle, Tristan S L'Ecuyer, Ann Kristin Naumann, Hauke Schulz, Pier A Siebesma, Jessica Vial, Dave M. Winker, and Paquita Zuidema. Sugar, gravel, fish and flowers: Mesoscale cloud patterns in the trade winds. *Quarterly Journal of the Royal Meteorological Society*, 146(726):141–152, 2020. doi: 10.1002/qj.3662.

[154] Bjorn Stevens, Sandrine Bony, Hélène Brogniez, Laureline Hentgen, Cathy Hohenegger, Christoph Kiemle, Tristan S. L'Ecuyer, Ann Kristin Naumann, Hauke Schulz, Pier A. Siebesma, Jessica Vial, Dave M. Winker, and Paquita Zuidema. Sugar, gravel, fish and flowers: Mesoscale cloud patterns in the trade winds. *Quarterly Journal of the Royal Meteorological Society*, 146(726):141–152, 2020. doi: 10.1002/qj.3662.

[155] Masanori Suganuma, Mete Ozay, and Takayuki Okatani. Exploiting the potential of standard convolutional autoencoders for image restoration by evolutionary search. In *35th International Conference on Machine Learning*, pages 7592–7601, 2018.

[156] Baris Sumengen, Anand Rajagopalan, Gui Citovsky, David Simcha, Olivier Bachem, Pradipta Mitra, Sam Blasiak, Mason Liang, and Sanjiv Kumar. Scaling hierarchical agglomerative clustering to billion-sized datasets. *arXiv preprint arXiv:2105.11653*, 2021.

[157] Karl E Taylor, Ronald J Stouffer, and Gerald A Meehl. An overview of cmip5 and the experiment design. *Bulletin of the American meteorological Society*, 93(4):485–498, 2012. doi: 10.1175/BAMS-D-11-00094.1.

[158] Mark Taylor, Peter M Caldwell, Luca Bertagna, Conrad Clevenger, Aaron Donahue, James Foucar, Oksana Guba, Benjamin Hillman, Noel Keen, Jayesh Krishna, et al. The simple cloud-resolving e3sm atmosphere model running on the frontier exascale system. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–11, 2023.

[159] Bridget Thrasher, Edwin P Maurer, Cheryl McKellar, and Philip B Duffy. Technical note: Bias correcting climate model simulated daily temperature extremes with quantile mapping. *Hydrology and Earth System Sciences*, 16(9):3309–3314, 2012. doi: https://doi.org/10.5194/hess-16-3309-2012.

[160] Bin Tian, M. K. Shaikh, M. R. Azimi-Sadjadi, T. H. Haar, and D. Reinke. A study of cloud classification with neural networks using spectral and textural features. *IEEE Transactions on Neural Networks*, 10:138–51, 1999. doi: 10.1109/72.737500.

[161] Xavier-Andoni Tibau, Christian Reimers, Christian Requena-Mesa, and Jakob Runge. Spatio-temporal autoencoders in weather and climate research. *Deep Learning for the Earth Sciences: A Comprehensive Approach to Remote Sensing, Climate Science, and Geosciences*, pages 186–203, 2021. doi: 10.1002/9781119646181.ch13.

[162] GN Triantafyllou and AA Tsonis. Assessing the ability of the köppen system to delineate the general world pattern of climates. *Geophysical Research Letters*, 21(25): 2809–2812, 1994. doi: 10.1029/94GL01992.

[163] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[164] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[165] Pascal Vincent, H. Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *International Conference on Machine Learning*, 2008. doi: 10.1145/1390156.1390294.

[166] Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *Proceedings of the 26th annual international conference on machine learning*, pages 1073–1080, 2009. doi: 10.1145/1553374.1553511.

[167] Ari Visa, Jukka Iivarinen, Kimmo Valkealahti, and Olli Simula. Neural network based cloud classifier. In *Industrial Applications of Neural Networks*, pages 303–309. World Scientific, 1998. doi: 10.1142/9789812816955_0035.

[168] Ulrike Von Luxburg. *Clustering Stability: An Overview*. Now Publishers Inc, 2010. doi: 10.48550/arXiv.1007.1075.

[169] Haruko M Wainwright, Sebastian Uhlemann, Maya Franklin, Nicola Falco, Nicholas J Bouskill, Michelle E Newcomer, Baptiste Dafflon, Erica R Siirila-Woodburn, Burke J Minsley, Kenneth H Williams, et al. Watershed zonation through hillslope clustering for tractably quantifying above-and below-ground watershed heterogeneity and functions. *Hydrology and Earth System Sciences*, 26(2):429–444, 2022. doi: 10.5194/hess-26-429-2022.

[170] Jiali Wang, Zhengchun Liu, Ian Foster, Won Chang, Rajkumar Kettimuthu, and V Rao Kotamarthi. Fast and accurate learned multiresolution dynamical downscaling for precipitation. *Geoscientific Model Development*, 14(10):6355–6372, 2021.

[171] Lijing Wang, Takuya Kurihana, Aurelien Meray, Ilijana Mastilovic, Satyarth Praveen, Zexuan Xu, Milad Memarzadeh, Alexander Lavin, and Haruko Wainwright. Multi-scale digital twin: Developing a fast and physics-informed surrogate model for groundwater contamination with uncertain climate models. *arXiv preprint arXiv:2211.10884*, 2022.

[172] Joe H Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, 1963. doi: 10.1080/01621459.1963.10500845.

[173] Michael S. Warren, Steven P. Brumby, Samuel W. Skillman, Tim Kelton, Brendt Wohlberg, Mark Mathis, Rick Chartrand, Ryan Keisler, and Mark Johnson. Seeing the earth in the cloud: Processing one petabyte of satellite imagery in one day. In *2015 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pages 1–12, 2015. doi: 10.1109/AIPR.2015.7444536.

[174] Duncan Watson-Parris, Yuhan Rao, Dirk Olivié, Øyvind Seland, Peer Nowack, Gustau Camps-Valls, Philip Stier, Shahine Bouabid, Maura Dewey, Emilie Fons, et al. Climatebench v1. 0: A benchmark for data-driven climate projections. *Journal of Advances in Modeling Earth Systems*, 14(10):e2021MS002954, 2022. doi: 10.1029/2021MS002954.

[175] R. Welch, S. K. Sengupta, and D. W. Chen. Cloud field classification based upon high spatial resolution textural features: 1. Gray level co-occurrence matrix approach. *Journal of Geophysical Research: Atmospheres*, 93(D10):12663–12681, 1988. doi: 10.1029/JD093iD10p12663.

[176] R. Welch, S. Sengupta, A. Goroch, P. Rabindra, N. Rangaraj, and M. Navar. Polar cloud and surface classification using AVHRR imagery: An intercomparison of methods. *Journal of Applied Meteorology*, 31(5):405–420, 1992. doi: 10.1175/1520-0450(1992)031⟨0405:PCASCU⟩2.0.CO;2.

[177] Denis S Willett, Jonathan Brannock, Jenny Dissen, Patrick Keown, Katelyn Szura, Otis B Brown, and Adrienne Simonson. Noaa open data dissemination: Petabyte-scale earth system data in the cloud. *Science Advances*, 9(38):eadh0032, 2023.

[178] R. Wood and D. Hartmann. Spatial variability of liquid water path in marine low cloud: The importance of mesoscale cellular convection. *Journal of Climate*, 19:1748–1764, 2006. doi: 10.1175/JCLI3702.1.

[179] Robert Wood. Stratocumulus clouds. *Monthly Weather Review*, 140(8):2373–2423, 08 2012. doi: 10.1175/MWR-D-11-00121.1.

[180] World Meteorological Organization. International Cloud Atlas. https://cloudatlas.wmo.int/.

[181] Xiaofu Wu, Suofei Zhang, Quan Zhou, Zhen Yang, Chunming Zhao, and Longin Jan Latecki. Entropy minimization versus diversity maximization for domain adaptation. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–12, 2021. doi: 10.1109/TNNLS.2021.3110109.

[182] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742, 2018.

[183] Wanyi Xie, Dong Liu, Ming Yang, Shaoqing Chen, Benge Wang, Zhenzhu Wang, Yingwei Xia, Yong Liu, Yiren Wang, and Chaofan Zhang. Segcloud: A novel cloud image segmentation model using a deep convolutional neural network for ground-based all-sky-view camera observation. *Atmospheric Measurement Techniques*, 13(4):1953–1961, 2020. doi: 10.5194/amt-13-1953-2020.

[184] You Xie, Erik Franz, Mengyu Chu, and Nils Thuerey. tempogan: A temporally coherent, volumetric gan for super-resolution fluid flow. *ACM Transactions on Graphics (TOG)*, 37(4):1–15, 2018.

[185] 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 *Natural Language Processing and Chinese Computing: 8th CCF International Conference, NLPCC 2019, Dunhuang, China, October 9–14, 2019, Proceedings, Part II 8*, pages 563–574. Springer, 2019. doi: 10.1007/978-3-030-32236-6_51.

[186] Z. Xu, R. Serata, H. Wainwright, M. Denham, S. Molins, H. Gonzalez-Raymat, K. Lipnikov, J. D. Moulton, and C. Eddy-Dilek. Reactive transport modeling for supporting climate resilience at groundwater contamination sites. *Hydrology and Earth System Sciences*, 26(3):755–773, 2022. doi: 10.5194/hess-26-755-2022. URL https://hess.copernicus.org/articles/26/755/2022/.

[187] Liang Ye, Z. Cao, and Yang Xiao. DeepCloud: Ground-based cloud image categorization using deep convolutional features. *IEEE Transactions on Geoscience and Remote Sensing*, 55:5729–5740, 2017. doi: 10.1109/TGRS.2017.2712809.

[188] Tianle Yuan, Hua Song, Robert Wood, Johannes Mohrmann, Kerry Meyer, Lazaros Oreopoulos, and Steven Platnick. Applying deep learning to nasa modis data to create a community record of marine low-cloud mesoscale morphology. *Atmospheric Measurement Techniques*, 13(12):6989–6997, 2020. doi: 10.5194/amt-13-6989-2020.

[189] Valentina Zantedeschi, Fabrizio Falasca, Alyson Douglas, Richard Strange, Matt Kusner, and Duncan Watson-Parris. Cumulo: A dataset for learning cloud classes. In *NeurIPS Workshop on Tackling Climate Change with Machine Learning*, 2019. https://www.climatechange.ai/papers/neurips2019/11.

[190] Mykhaylo Zayats, Małgorzata J. Zimoń, Kyongmin Yeo, and Sergiy Zhuk. Super resolution for turbulent flows in 2d: Stabilized physics informed neural networks. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 3377–3382, 2022. doi: 10.1109/CDC51059.2022.9992729.

[191] Mark D Zelinka, Timothy A Myers, Daniel T McCoy, Stephen Po-Chedley, Peter M Caldwell, Paulo Ceppi, Stephen A Klein, and Karl E Taylor. Causes of higher climate sensitivity in cmip6 models. *Geophysical Research Letters*, 47(1):e2019GL085782, 2020. doi: doi.org/10.1029/2019GL085782.

[192] J. Zhang, Pu Liu, Feng Zhang, and Qianqian Song. CloudNet: Ground-based cloud classification with deep convolutional neural network. *Geophysical Research Letters*, 45:8665–8672, 2018. doi: 10.1029/2018GL077787.

[193] Mingjin Zhang, Qianqian Wu, Jing Zhang, Xinbo Gao, Jie Guo, and Dacheng Tao. Fluid micelle network for image super-resolution reconstruction. *IEEE Transactions on Cybernetics*, 53(1):578–591, 2023. doi: 10.1109/TCYB.2022.3163294.

[194] Mingjin Zhang, Qianqian Wu, Jie Guo, Yunsong Li, and Xinbo Gao. Heat transfer-inspired network for image super-resolution reconstruction. *IEEE Transactions on Neural Networks and Learning Systems*, 35(2):1810–1820, 2024. doi: 10.1109/TNNLS.2022.3185529.

[195] Richard Zhang. Making convolutional networks shift-invariant again. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7324–7334. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/zhang19a.html.

[196] Xianliang Zhang and Xiaodong Yan. Spatiotemporal change in geographical distribution of global climate types in the context of climate warming. *Climate dynamics*, 43(3):595–605, 2014. doi: 10.1007/s00382-013-2019-y.