THE UNIVERSITY OF CHICAGO

TOWARDS THE DEMOCRATIZATION OF LARGE CLIMATE DATASET VIA UNSUPERVISED DEEP LEARNING TECHNIQUES

A DISSERTATION PROPOSAL SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE PHYSICAL SCIENCE
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF COMPUTER SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

BY
TAKUYA KURIHANA

CHICAGO, ILLINOIS
MAY, 2023

Table of Contents

CHAPTER 1

MOTIVATION AND INTRODUCTION

1.1 Research Question

This thesis aims to answer the following driving research question:

Can unsupervised cloud clustering systems allow the discovery of unknown climate-relevant cloud patterns, and democratize a large amount of climate data by accessing the core data more accessible?

1.2 Thesis statement

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. In climate science, multispectral satellite remote sensing instruments have provided petabytes of global multispectral cloud imagery over the past decades that capture cloud structure, size distributions, and radiative properties at a near-daily cadence. These observations should help in understanding cloud responses and trends in cloud behavior in climate science, whereas these enormous datasets are underutilized because climate scientists cannot in practice manually examine them to analyze spatial-temporal patterns. To aid the challenge, automated cloud classification algorithms using AI such as machine learning and deep learning have been studied to identify physically relevant cloud types. However, the use of satellite observations to understand cloud feedbacks in a warming climate has been hampered by the simplicity of existing cloud classification schemes, which are based on single–pixel cloud properties rather than utilizing spatial structures and texture, and based on a grid of classes defined by low, medium, or high values of cloud top pressure and optical thickness.

In this thesis, I develop automated cloud classification frameworks [1] to generate unique new cloud classes that detect meaningful distinctions between cloud textures, using only raw multispectral imagery as an input. That is, cloud classes are defined without reliance on location, time/season, derived physical properties, or pre-designated artificial class definitions. Having the data-driven cloud classification approach, I create a unique new AI-generated cloud dataset, which delivers the information from multi-spectral images in a compact form, enabling data-driven diagnosis of patterns of cloud organization. This thesis aims to help democratize climate research by developing unsupervised cloud clustering algorithms based upon substantial prior works [40, 41, 42, 43].

1.3 Introduction

Meteorologists have developed a variety of cloud classification schemes [75, 103], 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. In particular, 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 [98]. 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,

1. A note about terminology: As the literature is not consistent in making a distinction between cloud classification and cloud clustering, I use the two terms interchangeably in this thesis

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 [75, 76, 77]. 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 [103], 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 [102]. 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 [6, 45, 100, 101], and combined textual and cloud physical features [79, 80]. More recent work [48, 71, 86, 104, 105, 106, 107, 109] takes advantage of convolutional neural networks (CNNs) [39, 87] 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 [71]) 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 [20, 77], 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. [71] classified just four particular patterns of stratocumulus defined and manually labeled by Stevens et al. [90].

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 *unsupervised learning*, in which unknown patterns in data are learned without requiring predefined labels. The first demonstrations of unsupervised neural network methods applied to cloud images were made in the 1990s [93, 95]. Even with the primitive neural networks then available, Tian et al. [93] 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 [14] prototyped unsupervised cloud classification algorithms that used convolutional neural networks (CNNs) and produced cloud classes from the resulting compact representations via hierarchical agglomerative clustering (HAC) [33]. 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, unsupervised 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.

In an era where advances in both climate models [82] and satellite instruments create an increasing number of data at Petabytes scale in finer spatial and temporal resolution, the difficulty raises for extracting insight from complex patterns in such PB-scale datasets, and then furthermore for democratizing their rich information for users regardless of access to a large computation resource. Unsupervised DNNs often perform dimensionality reduction so as to map only relevant information from gigantic raw data into a compact lower-dimensional latent space for serving as a *Foundation Model* [61]. Specifically for the purpose of cloud classification, a standardized science product with AI-generated labels and associated cloud-related parameters may support studies of the response of clouds to forcing on timescales from hours to decades and to allow data-driven diagnosis of cloud organization and behavior and their evolution over time as CO_2 and temperatures increase in global warming.

1.4 Thesis Contribution

The primary goal of this thesis is to propose an unsupervised cloud classification framework and create a new unique AI-driven cloud label and physical properties dataset that 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. To achieve the democratization of large climate science data via cutting-edge deep learning techniques, the thesis proposes four solutions.

The primary contributions are:

- The development of a rotationally invariant (RI) autoencoder for extracting complex spatial patterns of cloud texture and radiances associated with cloud properties on a

lower dimensional latent representation. In particular, I develop a rotation-invariant loss function to make cloud images agnostic to the orientations.

- The design of a series of evaluation protocols to establish that resulting cloud clusters can be scientifically useful in assigning operational classifications to cloud images.

- Creation of a novel unsupervised learning-based climate dataset that composites 23 years of global AI-generated cloud classification atlas (AICCA). AICCA translates 872 TB of satellite images into 26.7 GB (patch-level Section 3.6.3) and 54 GB (daily-level Section 3.6.3) of class labels and cloud properties.

- A design of universal workflow to evaluate bias of simulated cloud from a high-resolution cloud model.

The thesis is organized as follows. Chapter 2 provides related work on unsupervised cloud classification, autoencoder, transform-invariant, and AI-generated climate datasets. In Chapter 3 I describe the proposed thesis contents. Chapter 4 presents initial results. item 5 describes the plan and timeline to complete thesis.

CHAPTER 2

RELATED WORK

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 cutting-edge neural network technologies after the mid-2010s. Most of the literature relies on supervised learning approaches. Although few unsupervised neural network approaches have been proposed since the 1990s [93, 95], early works arise limitations as the prediction accuracy in classification task falls behind that of supervised learning and the interpretation of resulting classes may not always match the standard cloud types. Recent algorithms and computing power advancements have motivated significant changes in cloud classification approaches. In particular, powerful graphics processing units (GPUs) have allowed researchers to train larger and more complex neural networks to tackle more sophisticated tasks. This chapter is dedicated to review literature in unsupervised cloud classification, autoencoders as a method of dimensionality-reduction, transform-invariant neural networks as a mean to perform more sophisticated tasks, and AI-driven climate datasets as a tool to help in the understanding of climate systems.

2.1 Unsupervised Neural Network approaches

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) [37]. 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 [93, 95] 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 [30], show that clusters produced using the SOM technique from ISCCP data [54, 55] and MODIS prodacts [83, 84] 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 [14] 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 [14], indicating a limitation in their ability to classify

small patches of large cloud structures by unsupervised learning.

2.2 Autoencoders

An autoencoder (AE) [24, 25, 38], 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 [21] 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 [78], denoising [94], and image inpainting [65, 91]. In addition, autoencoders have been applied

successfully to a wide variety of image recognition problems [9] via clustering and classification of the compact representation.

In current AI4science literature, autoencoder generally implies convolutional autoencoder (CAE) [49], 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 [39, 87]. 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. [68] 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 [3]. 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) [36] 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.

2.3 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. [53], Matsuo and Shimada [52] 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. [12] 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 [46] calculated the maximum of spherical cross-correlation for spherical 3D images in the loss function.

Shen et al. [85] 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. [8] 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 [11] 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. [29] discussed that Bootstrap Your Own Latent (BYOL) approach [17], 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. [15] 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 [31] 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 [88] 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 [110] subsamples feature maps from a naive max-pooling by a stride size, which achieves shit-invariance via a max pooling operation. Farabet et al. [16] transformed a raw input image through a Laplacian pyramid to obtain shift-invariant features. Kanazawa et al. [35] 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. [5] 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. [28] 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.

2.4 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 [19]. 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 [67, 81]. 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 [107] 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.

CHAPTER 3

PROPOSED THESIS CONTENT

The proposed approach uses multispectral satellite imagery from NASA's Moderate Resolution Imaging Spectroradiometer (MODIS) [34] 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) [57] and 2000 (Terra) [56] 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 Aqua band 6 [69]. The use of bands 5, 6, and 7 are only limited to local daylight times, which

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

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 × 128 pixels × 6 band region subsampled from a 2030 pixel × 1354 pixel × 36 band MODIS image. The total number of swath images per band is (12 swath/hour) × (12 hour/day) × (365 day/year) × (21 + 23 years, for Aqua and Terra, respectively) ≈ 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**) [7, 66] 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 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

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	§3.6
MOD06	Cloud mask		Cloud pixel detection	§3.2
	Land / Water		Background detection	
	Cloud optical thickness		Thickness of cloud	§3.5 & §3.6
	Cloud top pressure		Pressure at cloud top	
	Cloud phase infrared		Cloud particle phase	
	Cloud effective radius		Radius of cloud droplet	

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 [24, 25] 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.

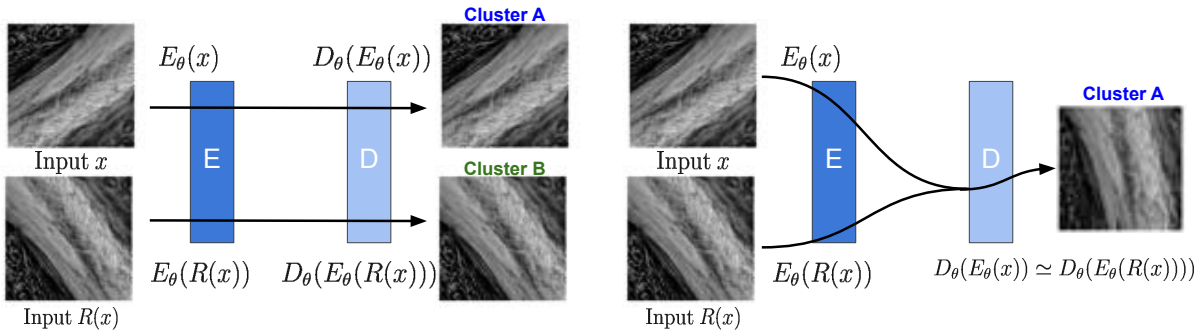
The loss function minimizes the difference between an original and a restored image based on a distance metric during autoencoder training. The most commonly used metric is a simple ℓ^2 distance between the autoencoder's input and output:

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

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.

To address this rotation-dependence problem in autoencoder, we propose a rotation-invariant (RI) loss function that generates similar latent representations, agnostic to orientation,



(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.

for similar morphological clouds (Figure 3.1b). This RI autoencoder, motivated by the shifted transform invariant autoencoder of Matsuo et al. [53], 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}:

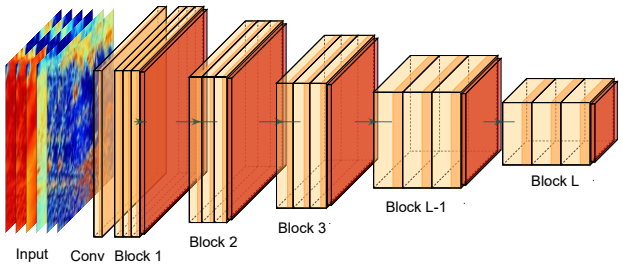
$$L_{\text{inv}}(\boldsymbol{\theta}) = \frac{1}{N} \sum_{x \in S} \sum_{R \in \mathcal{R}} ||D_\theta(E_\theta(x)) - D_\theta(E_\theta(R(x)))||_2^2, \tag{3.3}$$

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 [60], and with batch normalization [27]

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 [44] 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) [33] to merge pairs of clusters from bottom to top, minimizing at each step the linkage distance among merging clusters. We use Ward's method [97], 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++ [2], 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,

$$d(C_A, C_B) = E(C_A \cup C_B) - E(C_1) - E(C_2), \tag{3.5}$$

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 Evaluation Protocol

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 [75] 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 [23] 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 [62] 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) [47] 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 [70] 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) [26] 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 [96], 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 4.9c), 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.

3.6 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 3.6.1) designed to generate 42 clusters that are configured via the evaluation protocol (see Section 3.5).

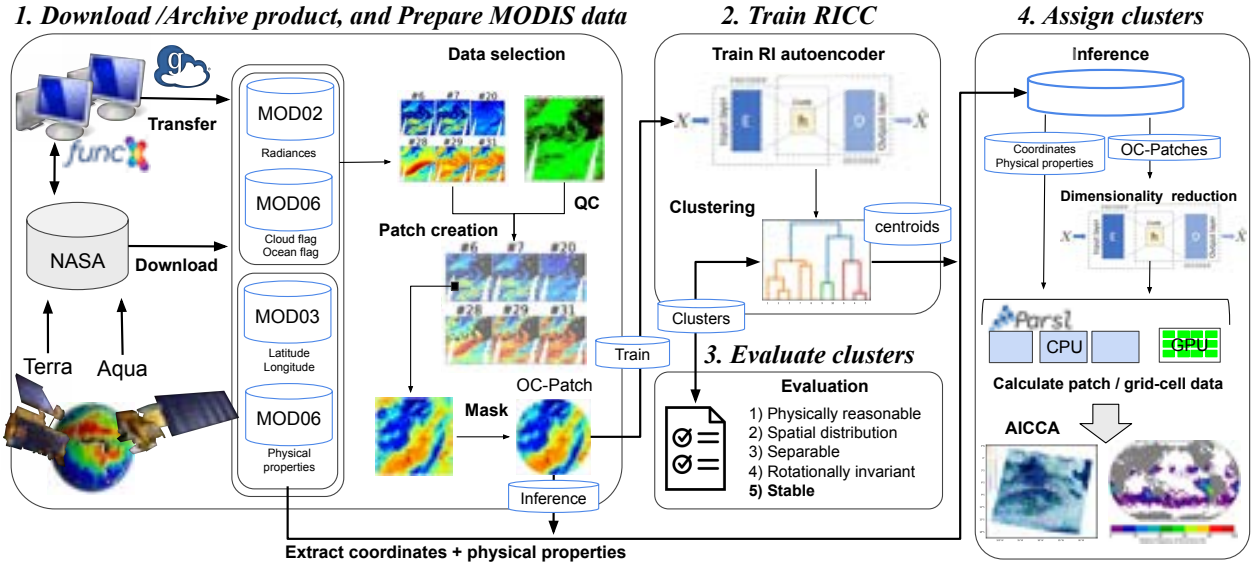


Figure 3.3: The AICCA production workflow comprises four principal stages. **1) Download/Archive and Prepare MODIS data**: Download calibrated and retrieved MODIS products from the NASA Level-1 and Atmosphere Archive and Distribution System (LAADS), using FuncX and Globus for rapid and reliable retrieval 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.

3.6.1 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 [58]—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 [32, 59, 92] 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:

$$c_{k,i} = \underset{k=\{1,\cdots,k^*\}}{\arg\min} ||z(x_i) - \mu_k||_2. \tag{3.16}$$

This label prediction or *inference* process is easily parallelized. We use the Parsl parallel Python library [4], which enables scalable execution on many processors via simple Python decorators, for this purpose.

3.6.2 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 3.3):

- `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 3.3), 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 3.3: 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 [72] 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.

3.6.3 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 3.6.2, 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 3.4, 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 3.4: 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 [64] and Dask [74] 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 × (365 days/year) × (12 000 patches/day) ×314 B/patch ≈ 56 gigabytes.

3.7 SCuBA: Self-supervised Cloud Bias Assessment

Finally the thesis proposal proposes Self-supervised Cloud Bias Assessment (SCuBA) framework that validates biases included in simulated clouds from high-resolution numerical climate simulations. Advances in computing power for HPC systems greatly encourage to scale 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) [89]. However, the state of the art 3–5 km simulations are not fully resolving deep convection, leading to biases in simulated clouds. Given that the clouds play the one of the largest uncertainty in climate projections [108], an 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.

The SCuBA framework may have potential advantages over existing supervised learning based model bias scheme [63] without reliance of limit location, time/season, and selection of climate models by employing a trained RI autoencoder to produce latent representations and using the class assignment scheme described in Section 3.6.1 to assign cluster labels for patches generated from climate models. 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. 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.

For variables from outputs of climate model, let ϕ be selected variables to be used for a satellite simulator M. Here, we use the Satellite Data Simulator Unit (SDSU) [50]. As high-resolution models include excessive number of layers where clouds are rarely observed < 100 hPa (e.g., polar mesospheric clouds), and layers near the tropopause that halts further upward motion of clouds, we select 45 model vertical layers especially more frequently from 700 to 1000 hPa altitude (line 1). To account for variables distributed non-continuously in the vertical direction, we calculate weighted mean (line 6) across layers using Gaussian function (line 5). Having pre-processed model variables, we execute satellite simulator to generate synthesized MODIS radiances for bands 6, 7, 20, 28, 29, and 31 and subsequently

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

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

1: $Z := \{z \mid z \in Z_{\mathrm{N}}\}$ ▷ Select vertical layer used for simulator
2: **for** j from 1 to $|\phi|$ **do**
3: Average a variable across the layers.
4: **for** z_i from Z **do**
5: $w(z) = \frac{1}{\sigma\sqrt{2\pi}} \exp -\frac{(z-z_i)^2}{2\sigma^2}$ where $\sigma = \sqrt{\frac{\sum(z-z_i)}{z_{i+1}-z_{i-1}}}$, ▷ Determine weights per layer
6: $\bar{\phi}_j^i = \frac{\int w(z)\cdot\phi_j^i \, dz}{\int w(z) \, dz}$
7: **end for**
8: **end for**
9: $\tilde{X} = M(\bar{\phi})$ where M : satellite simulator ▷ Synthesize MODIS radiances via simulator
10: **for** $\tilde{x_i}$ from $\tilde{X}$ **do**
11: $\tilde{c}_{k,i} = \underset{k=\{1,\cdots,42\}}{\arg\min} ||z(\tilde{x}_i) - \mu_k||_2$ where μ: centroids ▷ Assign cluster label for the i-th patch based on synthesized radiances
12: **end for**
13: Return cluster distribution of $\tilde{C}$

patches, giving $\tilde{X}$. We then assign one of AICCA class labels (line 11) via a trained RI autoencoder, giving a distribution of class labels $\tilde{C}$ to compare ones from AICCA dataset from real MODIS observation.

3.7.1 Project Plan

First, we develop SCuBA framework on a coarse-resolution model. As a preliminary test, we work with the eXperimental System for High-resolution prediction on Earth-to-Local Domains (X-SHiELD) [13] developed at the Geophysical Fluid Dynamics Laboratory (GFDL). We select the Satellite Data Simulator Unit (SDSU) [50] as a satellite simulator for generating MODIS radiances because SDSU employs a simple and computational efficient radiance calculation compared to other satellite simulators. We may deploy the Goddard Satellite Data Simulator Unit (G-SDSU) [51], an advanced version of SDSU, to achieve more complex and parallel node execution.

To develop and test SCuBA framework in a cloud-resolving scale model, we will work with the experimental nature run at global 1-km resolution (XNR1K) data from the global

1-km Integrated Forecast System (IFS) simulation [99] archived at Oak Ledge National Laboratory. XNR1K dataset provides global 1-km cloud and atmospheric data during the northern hemispheric winter months (NDJF: November to February) and the North Atlantic tropical cyclone season (ASO: August to October) at every 3 hours cadence. IFS model was initialized at 00 UTC on 1 November 2018 for NDJF, and 00 UTC on 1 August 2019 for ASO. Here, we plan using first seven days to validate synthesized radiances from SDSU simulator with real MODIS radiances. Simulated model variables after first seven days may deviate from real atmosphere, so we do not use for validation of radiances from satellite simulator but only use for inputs for SCuBA framework. Once we verify the performance of SDSU, we apply SCuBA workflow 3 to XNR1K's NDJF and then ASO data. As IFS data output provides ocean and land flag, we only work with simulated ocean clouds.

CHAPTER 4

INITIAL RESULTS

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).

4.1 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 4.1 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 4.1a 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 4.1b 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:

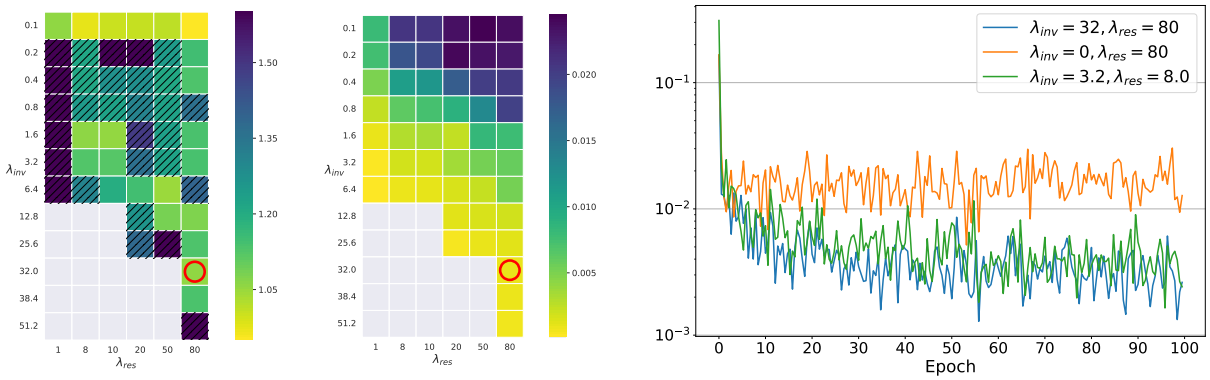
$$\text{Cosine similarity} = \frac{< X, X' >}{\|X\| \cdot \|X'\|}. \tag{4.1}$$

Equation (4.1) gives 1 when elements of two vectors are exactly matched. A standard deviation closer to 0 in Fig. 4.1b 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 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. 4.1c shows the evolution of training loss over iterations with different parameter values. The lowest convergence of the blue line in Fig. 4.1c 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.

4.2 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. [43] 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



(a) Ratio of restoration losses

(b) Similarity of restored images.

(c) Training losses for different λ combinations.

Figure 4.1: 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.

gain more generalized representations [10]. The other four models (Model–A – Model–D) are trained based on different initialization for 400 epochs. Figure 4.2 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 4.1 summarizes the number of GPUs used for training, the size of minibatch per GPU, and the total size of GPUs.

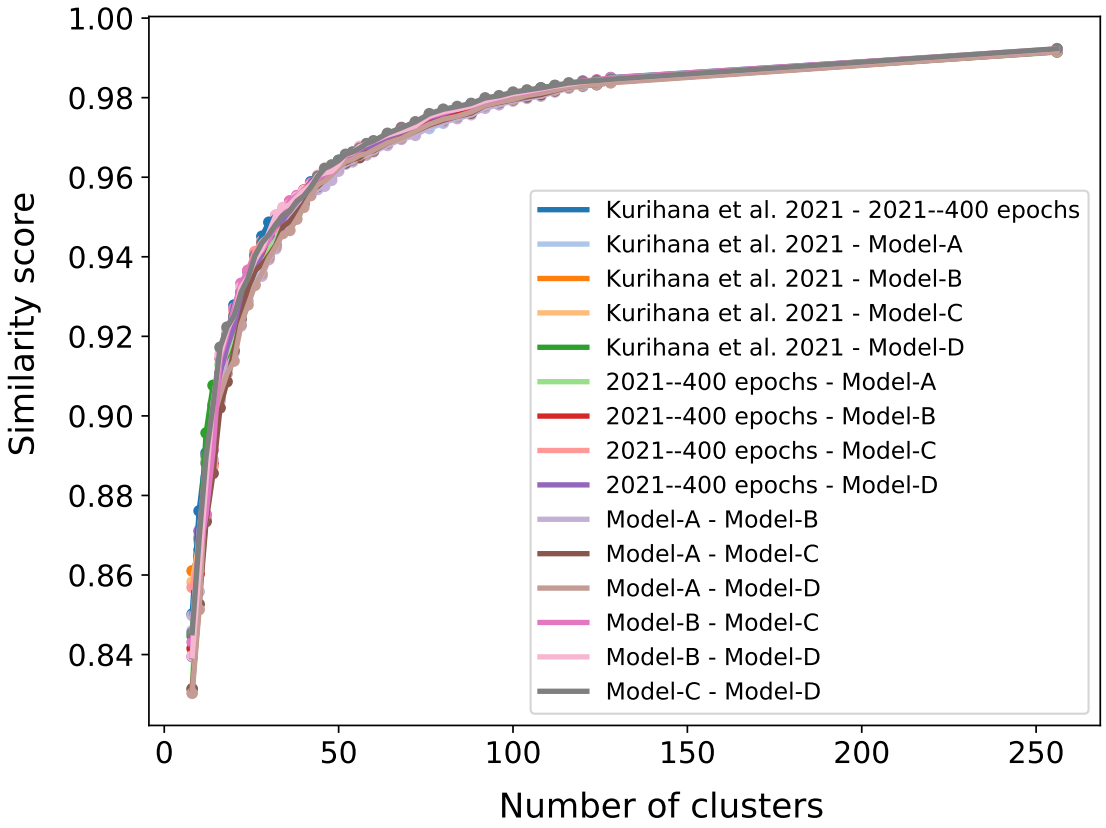


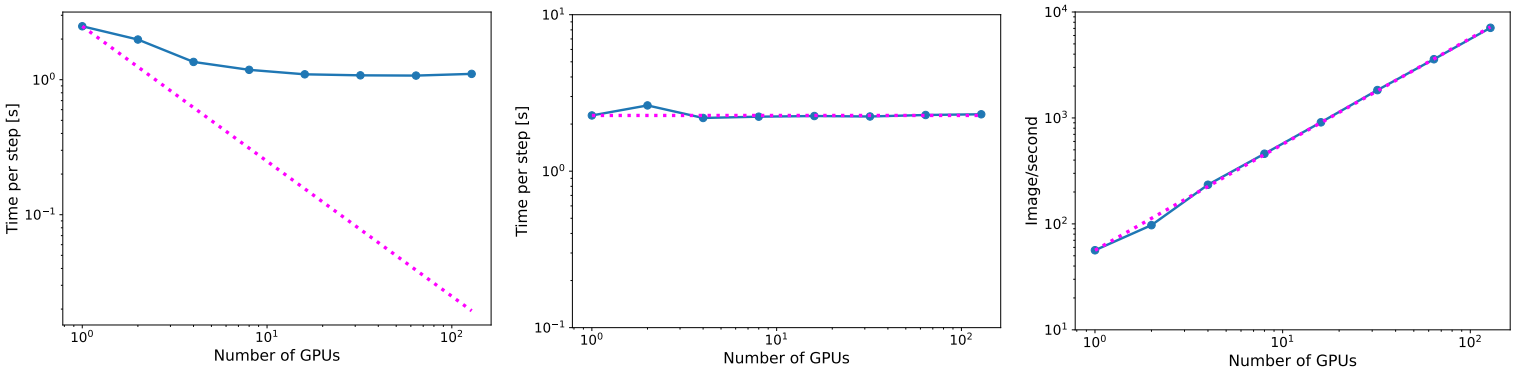
Figure 4.2: Plots of clustering agreement score (Rand Score Index) among six RI autoencoders based on $\texttt{OC-Patches}_{\texttt{HAC}}$ from 8 to 256 clusters. Agreement scores show similar ranges among models for the number of clusters tested, indicating that RI autoencoders are robust to generate similar latent representation and thus result in similar clustering assignments in particular for a larger number of clusters.

Table 4.1: 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 4.3 plots the results of strong and weak scaling based on a training time per step and throughput from weak scaling. Figure 4.3b indicates that the training of RI autoencoder efficiently leverages a larger size of GPUs under a weak scaling experiment by 128 GPUs, and the throughput line in Figure 4.3c matches with the ideal scaling curve. Instead, Figure 4.3a

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.



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

Figure 4.3: 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.

4.3 Evaluation protocol

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) autoencoder.

4.3.1 Physically Reasonable

Figure 4.4 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

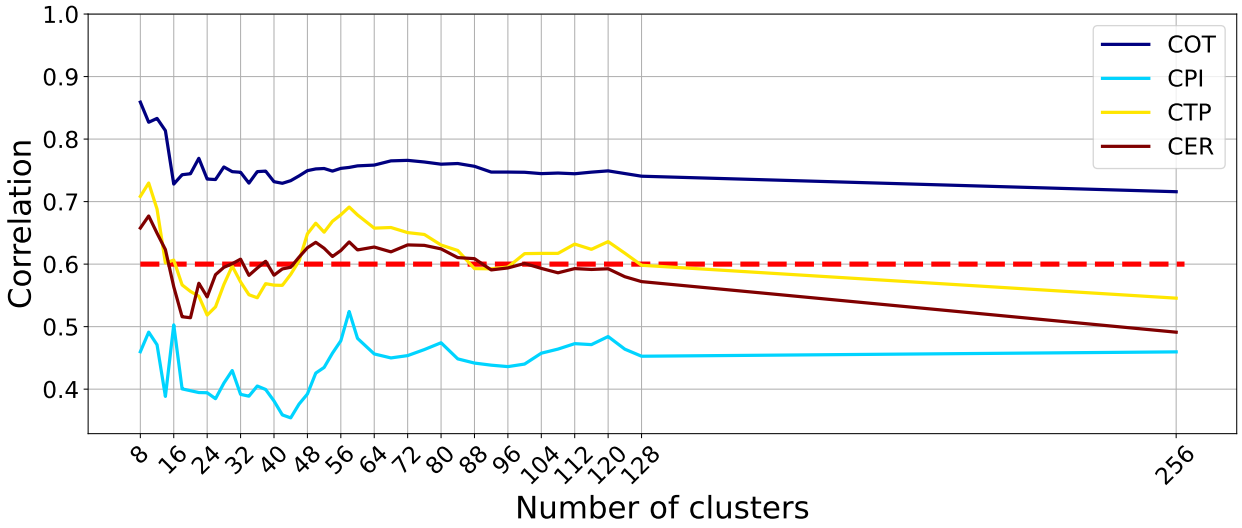


Figure 4.4: Plots of the median of correlation coefficients for four cloud physics parameters (COT, CPI, CTP, and CER) for RI autoencoders. Correlation coefficients <0.6 is a threshold for whether resulting clusters can group unique distributions of physical parameters. Three of the four variables fall below 0.6 from 16 to 46 and > 128 clusters, whereas the majority of clusters have highly similar optical thickness distributions.

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.

4.3.2 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 4.5 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.

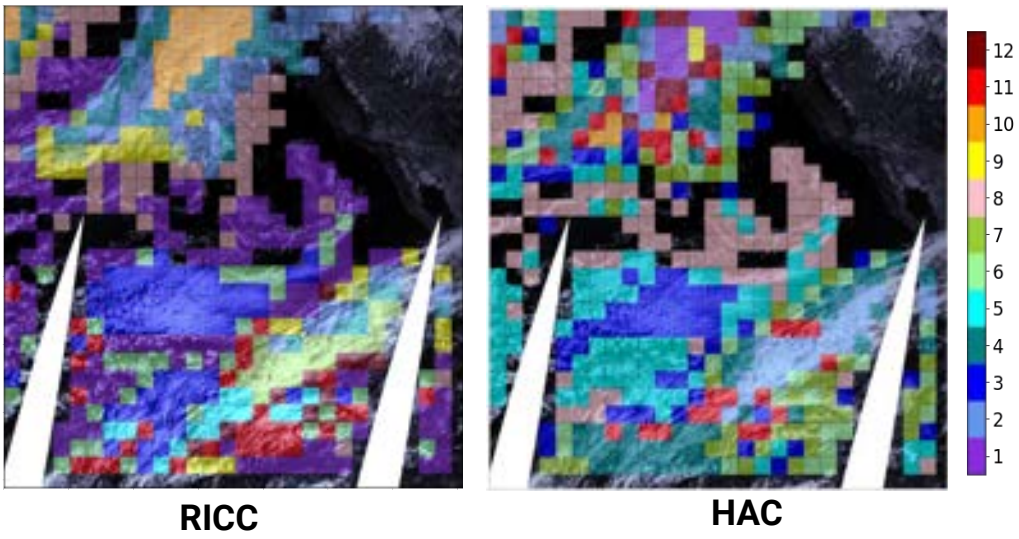
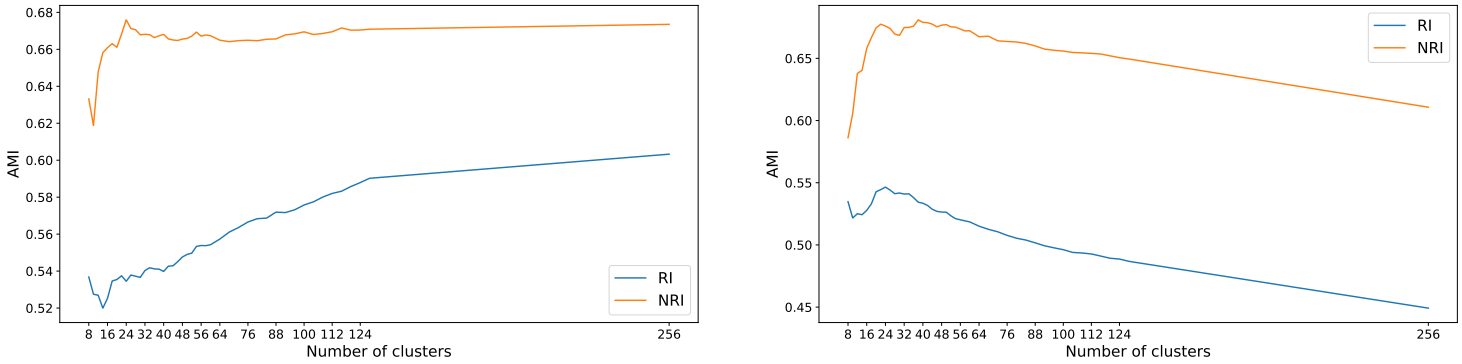


Figure 4.5: On the same MODIS swath with 493 patches, we show; Left: Clusters produced by RICC; Right: Clusters produced by HAC applied to patch-mean values of COT, CTP, CWP, and CER. The background raw visible image from MODIS band 1 is provided to show the context of geolocation and the presence of 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.

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 4.6a compares the results obtained from RI autoencoder and 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



(a) Smoothing

(b) Scrambling

Figure 4.6: 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.

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 4.6b 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 can be accounted for with and without rotation-invariance.

4.3.3 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.

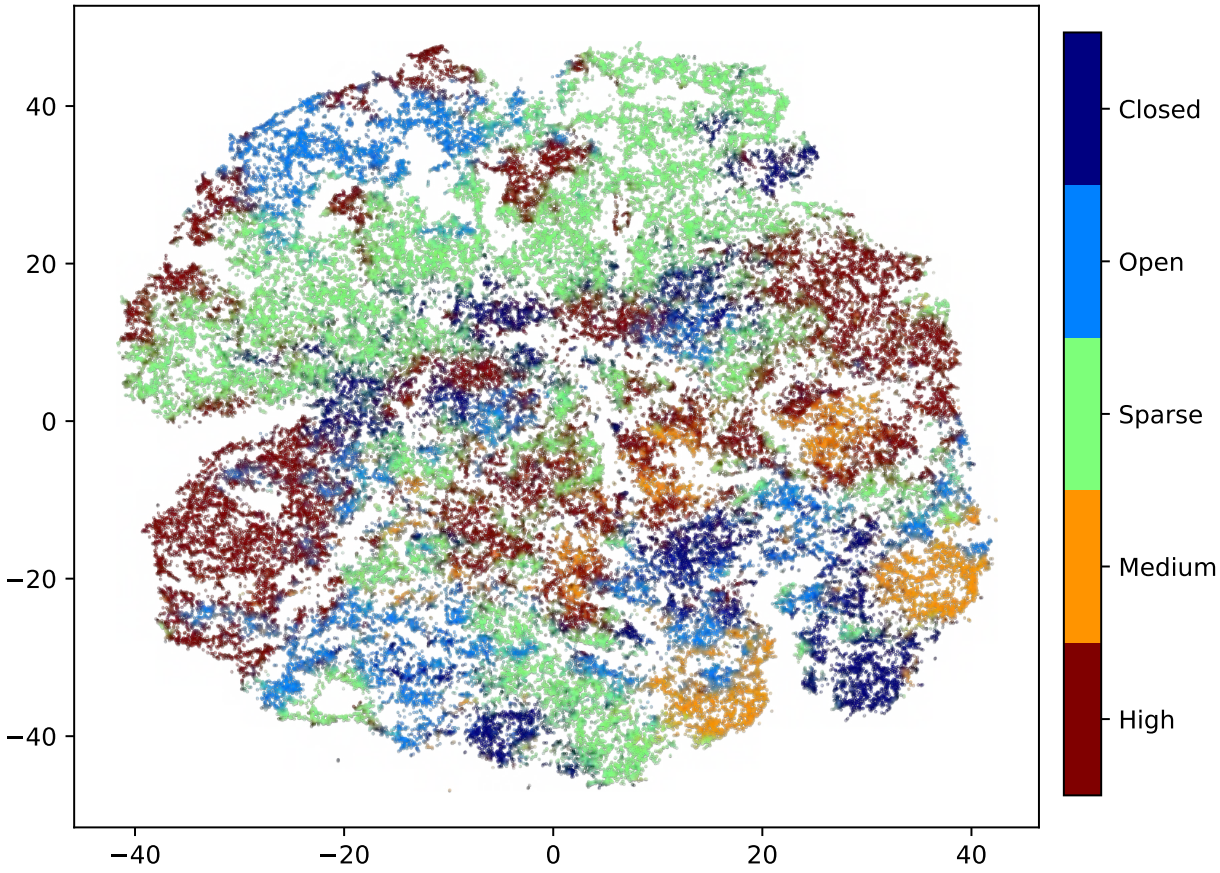


Figure 4.7: t-SNE visualization of latent representations of $\texttt{OC-Patches}_{\texttt{HAC}}$. To generalize the relationship between positions of patches and cluster labels in various cluster numbers k from 8 to 256, we regroup a set of clusters into *High* altitude, *Medium* altitude, *Sparse*, *Open* cell stratocumulus, and *Closed* cell stratocumulus clouds based on texture and within-cluster mean cloud optical thickness and cloud top pressure values.

Figure 4.7 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.

4.3.4 Rotation Invariance

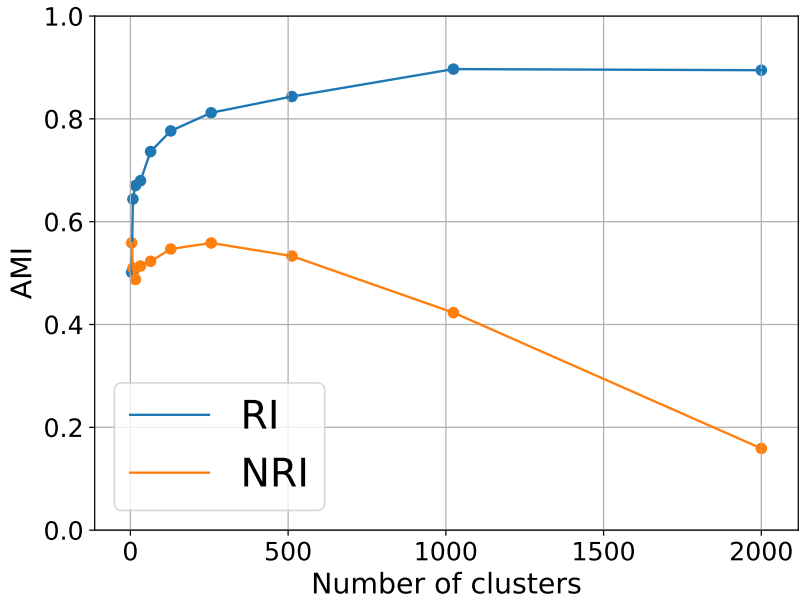


Figure 4.8: multi-cluster, applied to NRI and RI autoencoders. Clustering agreement scores (AMI) on the `Test` dataset, for from 4 to 2000 clusters. The AMI curve for RICC converges at 0.9, meaning that RICC autoencoder produces rotation-invariant latent representation.

Fig. 4.8 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.

4.3.5 Stability

Figure 4.9a 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 4.9b drops to 1.01 at 50 clusters, indicating that our optimal number of clusters is $k^* < 50$.

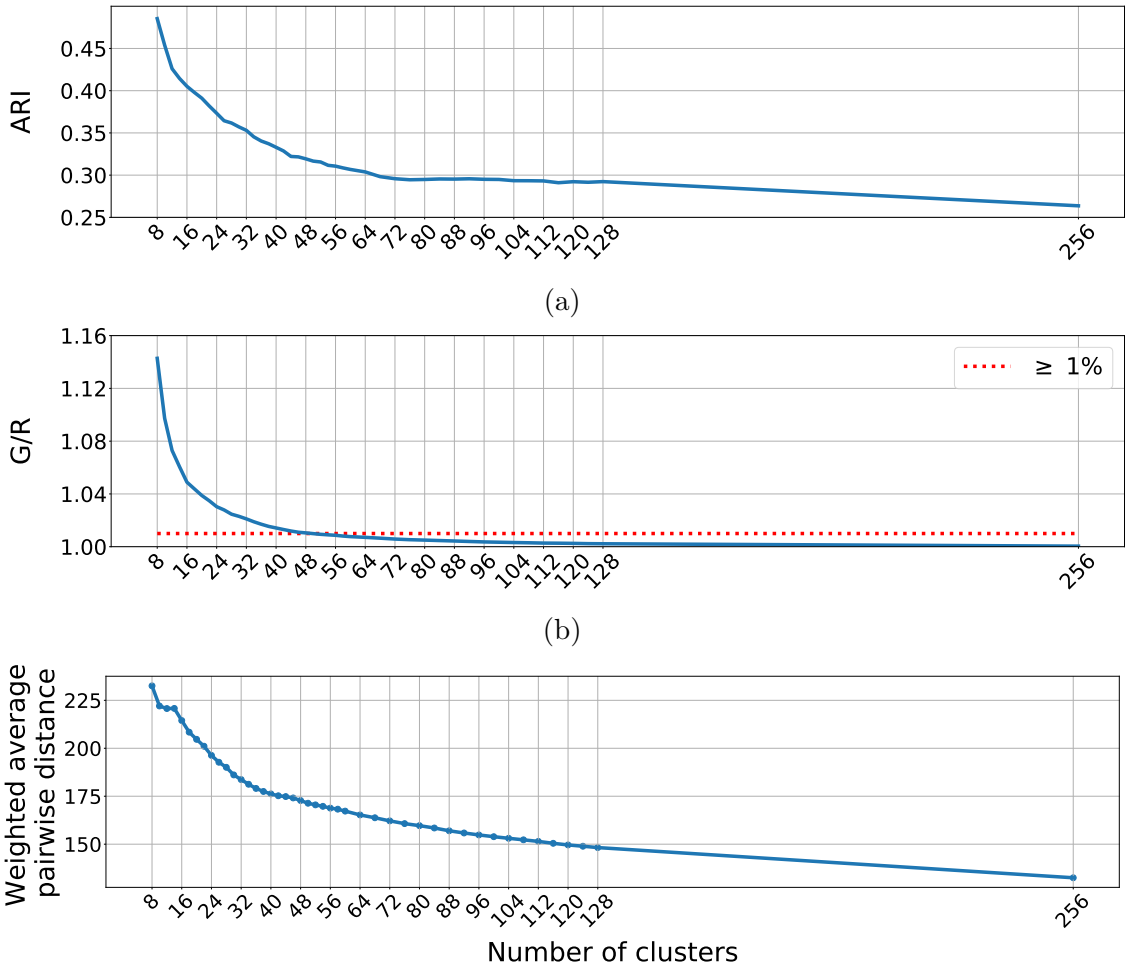


Figure 4.9: Plots for the three stability criteria metrics of Table 3.2, each as a function of number of clusters. (a) *Clustering similarity*: Adjusted Rand Index (ARI) as a measure of similarity of clusterings generated by RICC models trained on different subset of patches. (b) *Clustering similarity 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.

In Figure 4.9c, 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 lies in $40 \leq k^* \leq 48$.

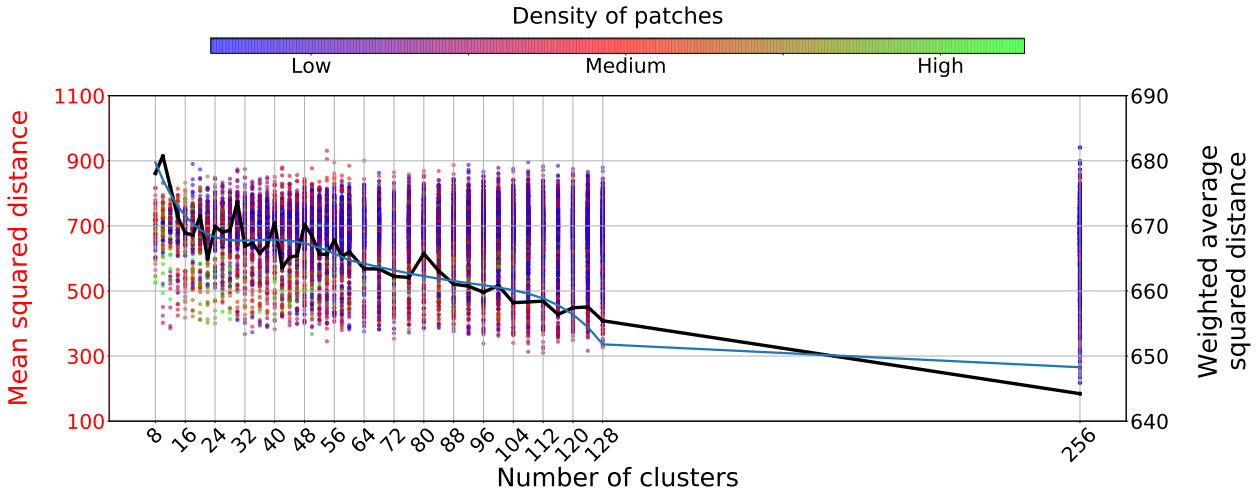


Figure 4.10: Seasonal stability test comparing the intra-seasonal variance of textures within each cluster as a function of number of clusters. Each of $9 \cdot k$ colored dots for each value of k gives the average squared distance (left y-axis) between July and January patches as described in text; 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.

These are shown as the dots in Figure 4.10. The WASD curve (black) decreases with 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 3.6.1. 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.

4.4 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.

4.4.1 Distribution of clusters

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 4.12 shows six true color images [18] 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.

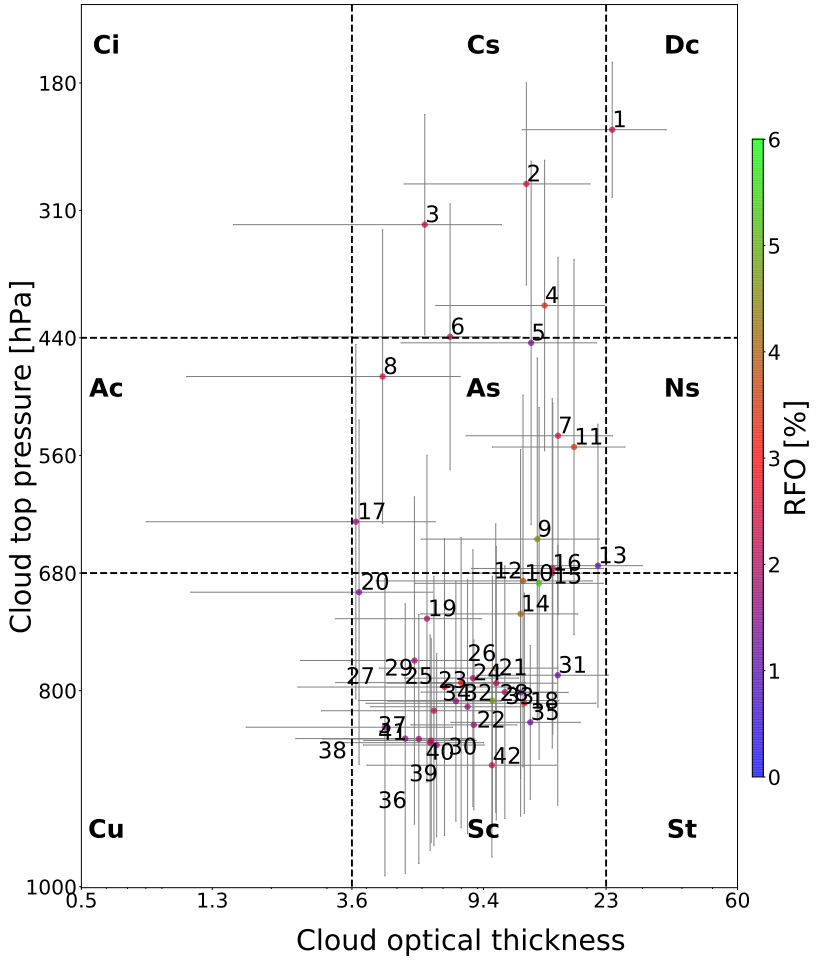
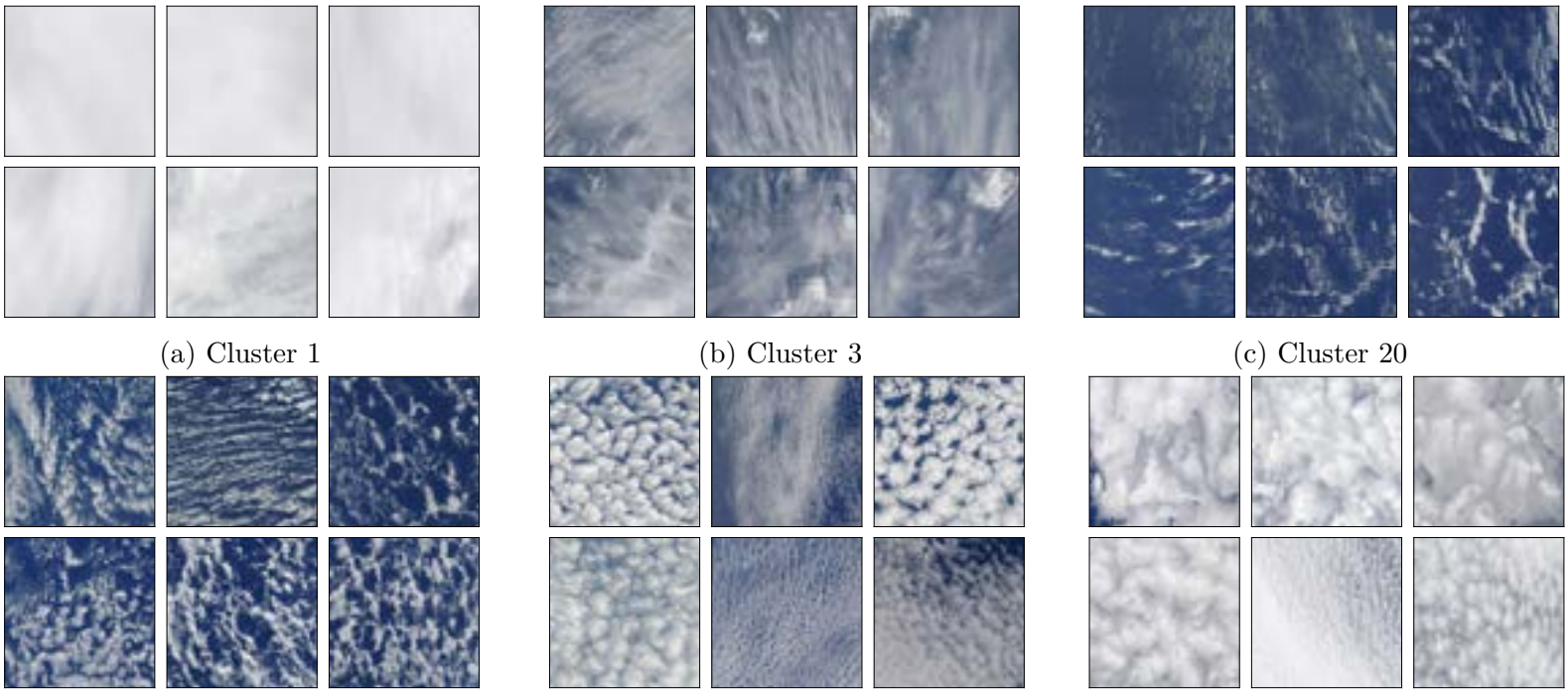


Figure 4.11: Distributions of cluster occurrences and properties from 2000 to 2021 for AICCA in COT–CTP space, where COT is cloud optical thickness (dimensionless) and CTP cloud top pressure (hPa). For comparison, dashed lines divide the nine regions corresponding to the International Satellite Cloud Climatology Project (ISCCP) cloud classes [76, 103]. 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.

4.4.2 Geographic Distribution of Cluster Label Occurrence

In this study, we leverage AICCA daily-level dataset (see Section 3.6.3) to examine the geographic distribution of AICCA cluster labels from 2000 to 2022. Figure 4.13 shows mean incidences for each of the 42 cloud types in the dataset, gridded on a 1° global



(a) Cluster 1 (b) Cluster 3 (c) Cluster 20

(d) Cluster 25 (e) Cluster 30 (f) Cluster 35

Figure 4.12: Selected MODIS true color images [18] 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 4.11.

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 3.6.1 for details. By contrast, the stratocumulus cloud labels discussed for Figure 4.11 show different 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 [73], 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.

4.4.3 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

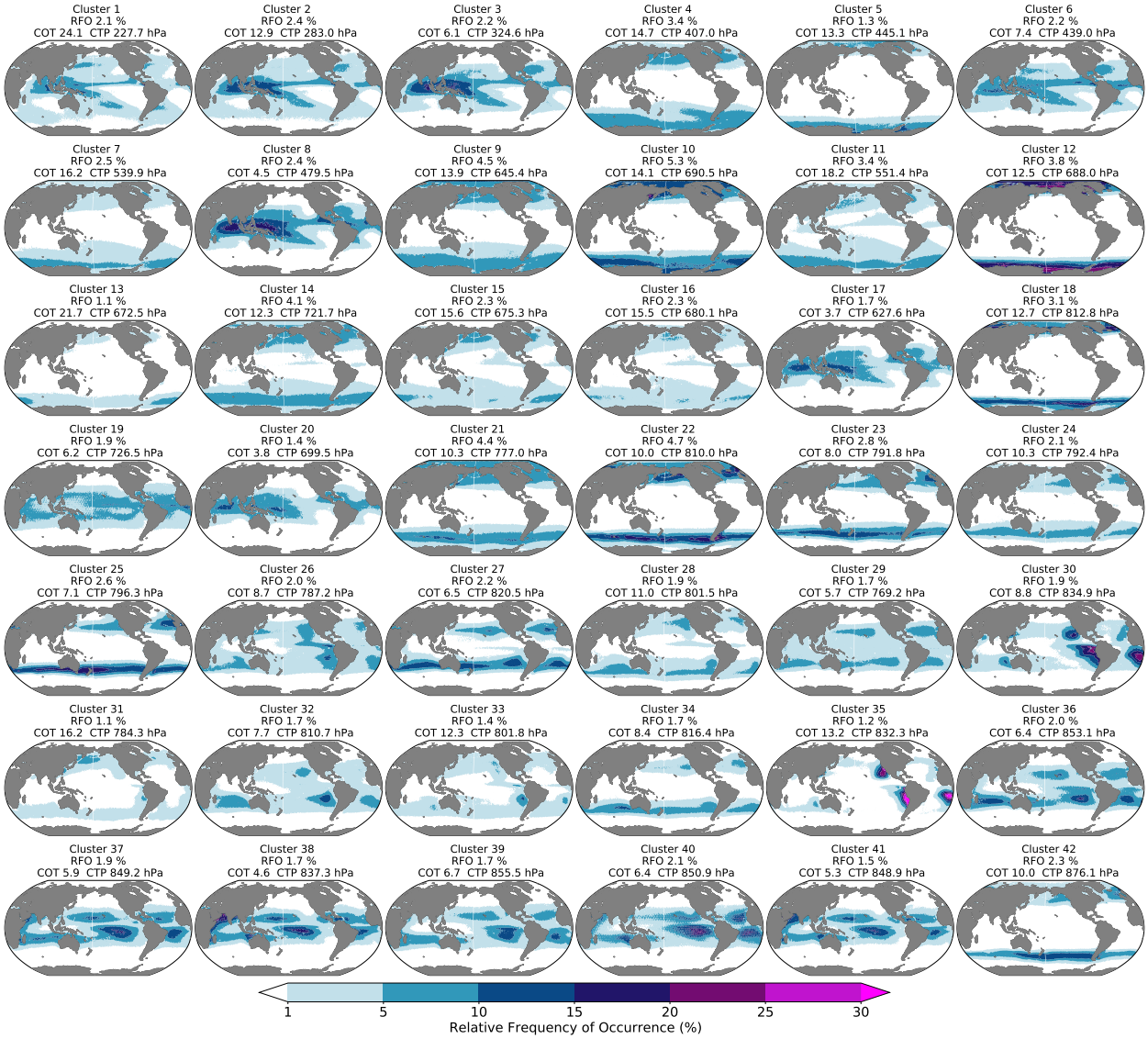


Figure 4.13: An example application of AICCA. We plot the relative frequency of occurrence (RFO) for each of the 42 AICCA_{42} clusters, using all data from 2000 to 2021. Land is in grey, and areas where RFO $< 1.0\%$ are in white. Surtitles show global mean RFO, cloud optical thickness (COT), and cloud 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 4.14 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.

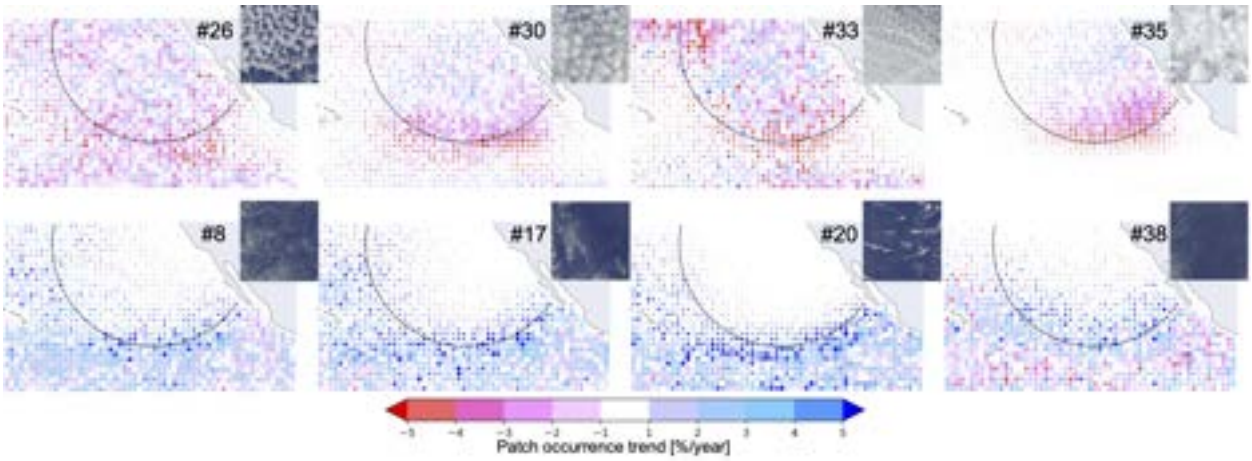


Figure 4.14: Trends in occurrence of selected AICCA cloud classes over the subtropical N. Pacific Ocean (100–160W and 5–40N). Colors show fitted linear trends over the 18 years 2003–2021, expressed in units of % of the mean value over this time. Dot size represents mean relative frequency of occurrence 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 [22] 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

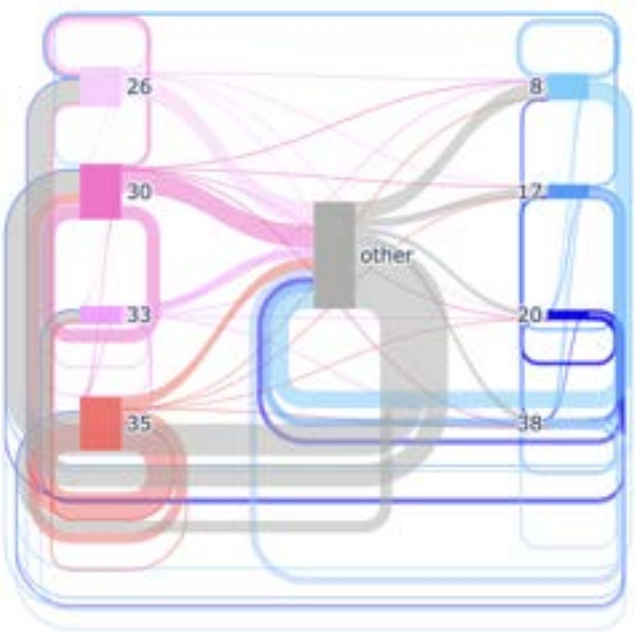


Figure 4.15: Sankey diagram of sub-daily and daily cloud class transitions within gridcells for the 2003–2022 period for the eight classes, and region, in Figure 4.14. Box widths represent occurrence frequency of each class, and colors are ordered by trend. Ribbon width represents the rate of transitions 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 4.15 shows, as a Sankey diagram, transitions from one time period to the next ($\sim$ daily) for the selected classes and region shown in Figure 4.14. 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 4.14 does not simply result from a reduction in stratocumulus lifetime.

4.5 SCuBA: Model comparison

We conduct analysis to compare the distribution of AICCA cloud classes from real MODIS observation between 2000–2022 and synthesized radiances from the eXperimental System for High-resolution prediction on Earth-to-Local Domains (X-SHiELD) [13] developed at the Geophysical Fluid Dynamics Laboratory (GFDL). We use the Satellite Data Simulator Unit (SDSU) [50] to obtain simulate radiances from the climate model. In a preliminary analysis, we use a one-day snapshot of X-SHiELD model output that is upscaled from 3.5 km to 6.5 km horizontal scale.

4.16 shows distributions of cluster frequencies for the entire 23-year AICCA dataset and those from the 6.5 km X-SHiELD model output. As expected, climate model outputs overestimate high clouds (#4 and #5) as well as low cloud classes (#37 and #38), and thus underestimating medium altitude clouds, indicating the model bias in terms of simulating clouds. Interestingly, a type of cloud class #37, which represents optically thin (COT 5.9 & CTP 849.2 hPa) low clouds, shows the highest frequency of occurrences (16.6%), even though #37 has only 1.9% of partition of patches based on MODIS observation for the entire 23 years. Results suggest that since the horizontal resolution of 6.5km does not reproduce the fine texture and structure of clouds, distributions of AICCA class based on synthesized radiances may contradict those from observations.

The preliminary result suggests that SCuBA framework can be used as a new diagnostic tool to evaluate biases from climate simulations, which is one of the largest uncertainties in the future climate projection. Because the satellite simulator is applicable to any climate models, our proposed tool shows a new direction of research in an application of unsupervised deep learning in climate science for the purpose of AI-based cloud bias correction.

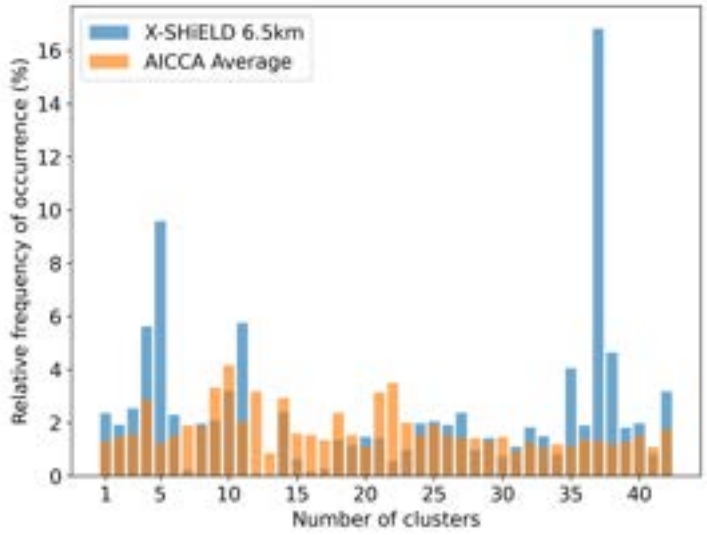


Figure 4.16: Histogram of distributions of 42 cloud classes in AICCA_{42} from the mean relative frequency of occurrences from 2000-2022 of `OC-Patches` , and those from synthesized radiances based on one day snapshot of X-SHiELD model 6.5km resolution.

CHAPTER 5

RESEARCH TIMELINE

Based on the proposed thesis content Chapter 3, I completed Data collection (Section 3.1), development of rotationally invariant autoencoder algorithm and the training protocol (Section 3.2 and Section 3.3), design of evaluation protocol under the condition of no ground-truth cloud labels are available (Section 3.5), and the creation of the novel AI-driven cloud classification atlas (AICCA) (Section 3.6). I plan to compile AICCA for the publicly available dataset (Section 3.6) by *Summer 2023*.

To complete the remaining part of the thesis in Section 3.7, I have to conduct the following steps:

Step-1 Collect the cloud-resolving scale (i.e., ideally spatial resolution $\leq$ 3 km) climate model outputs;

Step-2 Compile and adjust a satellite simulator code on the Midway cluster computer and the Argonne Leadership Computing Facility's Theta computer for the cloud-resolving scale inputs;

Step-3 Complete workflow to convert raw climate outputs to variables needed to feed the satellite simulator;

Step-4 Generate synthesized MODIS radiances from the variables being converted in the previous step; and

Step-5 Finally, finalize a workflow to validate biases of simulated clouds generated from climate models with the framework of distributions of cloud classes in AICCA.

For the tasks to be completed, I have already collaborated with Valentine Anantharaj in the Oak Ridge National Laboratory to obtain access permission to the experimental nature run at global 1-km resolution (XNR1K) data from the global 1-km Integrated Forecast System (IFS)

simulation [99], and computing resources from Oak Ridge Leadership Computing Facility. I will use the *Summer 2023* to run the experiments and perform validation tasks with XNR1K. I plan to submit these results to American Geophysical Union (AGU) Fall Meeting 2023 and/or journals that target AI applications to climate science.

Finally, I will use the *Autumn 2023* to write a draft of Ph.D. thesis and schedule a dissertation defense on the mid-December, 2023

I summarized the work plan as follows:

Spring 2023

- Collect XNR1K data from ORNL
- Prepare compilation and configuration to run satellite simulator

Summer 2023

- Generate synthesized MODIS radiances
- Conduct validation of simulated clouds from XNR1K data
- Submit an abstract to AGU 2023
- Make AICCA publicly available

Autumn 2023

- Finalize draft of the Ph.D. dissertation
- Defense on the mid-December

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

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

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

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

[10] 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.

[11] 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.

[12] 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.

[13] Kai-Yuan Cheng, Lucas Harris, Christopher Bretherton, Timothy M Merlis, Maximilien Bolot, Linjiong Zhou, Alex Kaltenbaugh, Spencer Clark, and Stephan Fueglistaler. Impact of warmer sea surface temperature on the global pattern of intense convection: insights from a global storm resolving model. *Geophysical Research Letters*, 49(16):e2022GL099796, 2022. doi: 10.1029/2022GL099796.

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

[15] 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.

[16] 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.

[17] 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.

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

[19] 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.

[20] 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.

[21] 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.

[22] 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`.

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

[24] 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.

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

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

[27] 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.

[28] 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.

[29] 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.

[30] 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`.

[31] 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.

[32] 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.

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

[34] 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.

[35] 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.

[36] 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`.

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

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

[39] 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.

[40] Takuya Kurihana, Ian Foster, Rebecca Willett, Sydney Jenkins, Kathryn Koenig, Ruby Werman, Ricardo Barros Lourenco, Casper Neo, and Elisabeth Moyer. Cloud classification with unsupervised deep learning. In *9th International Workshop on Climate Informatics*, 2019. doi: 10.48550/arXiv.2209.15585.

[41] 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`.

[42] Takuya Kurihana, James Franke, Ian Foster, Ziwei Wang, and Elisabeth Moyer. Insight into cloud processes from unsupervised classification with a rotationally invariant autoencoder. *arXiv preprint arXiv:2211.00860*, 2022. doi: 10.48550/arXiv.2211.00860.

[43] 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`.

[44] 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.

[45] 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.

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

[47] 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`.

[48] 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.

[49] 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.

[50] 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.

[51] T Matsui, J Santanello, JJ Shi, W-K Tao, D Wu, C Peters-Lidard, E Kemp, M Chin, D Starr, M Sekiguchi, et al. Introducing multisensor satellite radiance-based evaluation for regional earth system modeling. *Journal of Geophysical Research: Atmospheres*, 119(13):8450–8475, 2014. doi: 10.1002/2013JD021424.

[52] 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.

[53] 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.

[54] 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.

[55] 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.

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

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

[58] 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.

[59] 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.

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

[61] Tung Nguyen, Johannes Brandstetter, Ashish Kapoor, Jayesh K Gupta, and Aditya Grover. Climax: A foundation model for weather and climate. *arXiv preprint arXiv:2301.10343*, 2023.

[62] X. V. Nguyen, Jeffrey Chan, S. Romano, and J. Bailey. Effective global approaches for mutual information based feature selection. In *20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 512–521, 2014. doi: 10.1145/2623330.2623611.

[63] 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.

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

[65] 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.

[66] 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.

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

[68] 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.

[69] 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.

[70] 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.

[71] 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.

[72] 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.

[73] 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`.

[74] 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.

[75] 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.

[76] 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.

[77] 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.

[78] 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.

[79] 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`.

[80] 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.

[81] 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.

[82] Tapio Schneider, Colleen M Kaul, and Kyle G Pressel. Possible climate transitions from breakup of stratocumulus decks under greenhouse warming. *Nature Geoscience*, 12(3):163–167, 2019. doi: 10.1038/s41561-019-0310-1.

[83] 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.

[84] 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.

[85] 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.

[86] 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.

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

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

[89] 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. doi: 10.1186/s40645-019-0304-z.

[90] 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.

[91] 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.

[92] 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, 2021.

[93] 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.

[94] 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.

[95] 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.

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

[97] 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.

[98] 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.

[99] Nils P Wedi, Inna Polichtchouk, Peter Dueben, Valentine G Anantharaj, Peter Bauer, Souhail Boussetta, Philip Browne, Willem Deconinck, Wayne Gaudin, Ioan Hadade, et al. A baseline for global weather and climate simulations at 1 km resolution. *Journal of Advances in Modeling Earth Systems*, 12(11):e2020MS002192, 2020. doi: 10.1029/2020MS002192.

[100] 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.

[101] 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.

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

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

[104] 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.

[105] 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.

[106] 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.

[107] 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`.

[108] 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.

[109] 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.

[110] 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`.