Astronomy and Computing 49 (2024) 100884

Contents lists available at ScienceDirect

Astronomy and Computing

journal homepage: www.elsevier.com/locate/ascom

Full length article

Classifying the clouds of Venus using unsupervised machine learning

J. Mittendorf [a,b,*], K. Molaverdikhani [a,b,c], B. Ercolano [a,b,c], A. Giovagnoli [b,d], T. Grassi [c,e]

[a] University Observatory Munich, Faculty of Physics, LMU Munich, Scheinerstr. 1, Munich, D-81679, Germany
[b] Ludwig-Maximilians-Universität München, Geschwister-Scholl-Platz 1, Munich, D-80539, Germany
[c] Excellence Cluster ORIGINS, Boltzmannstr. 2, Garching, D-85748, Germany
[d] German Aerospace Center (DLR), Microwaves and Radar Institute, Münchener Straße 20, Weßling, D-82234, Germany
[e] Max Planck Institute for Extraterrestrial Physics, Giessenbachstr. 1, Garching, D-85748, Germany

ARTICLE INFO

Dataset link: https://github.com/jmittendo/planet-patch-classifier

Keywords:
Venus
Venus clouds
Venus atmosphere
Machine learning
Convolutional neural network

ABSTRACT

Because Venus is completely shrouded by clouds, they play an important role in the planet's atmospheric dynamics. Studying the various morphological features observed on satellite imagery of the Venusian clouds is crucial to understanding not only the dynamic atmospheric processes, but also interactions between the planet's surface structures and atmosphere. While attempts at manually categorizing and classifying these features have been made many times throughout Venus' observational history, they have been limited in scope and prone to subjective bias. We therefore present and investigate an automated, objective, and scalable approach for their classification using unsupervised machine learning that can leverage full datasets of past, ongoing, and future missions.

To achieve this, we introduce a novel framework to generate nadir observation patches of Venus' clouds at fixed consistent scales from satellite imagery data of the *Venus Express* and *Akatsuki* missions. Such patches are then divided into classes using an unsupervised machine learning approach that consists of encoding the patch images into feature vectors via a convolutional neural network trained on the patch datasets and subsequently clustering the obtained embeddings using hierarchical agglomerative clustering.

We find that our approach demonstrates considerable accuracy when tested against a curated benchmark dataset of Earth cloud categories, is able to identify meaningful classes for global-scale (3000 km) cloud features on Venus and can detect small-scale (25 km) wave patterns. However, at medium scales (~500 km) challenges are encountered, as available resolution and distinctive features start to diminish and blended features complicate the separation of well defined clusters.

1. Introduction

Venus is enveloped by a thick layer of clouds that are nearly featureless in visible light but show many variable features in the UV spectrum (e.g. Rossow et al., 1980), particularly around 365 nm – the characteristic wavelength of the unknown UV absorber (e.g. Molaverdikhani et al., 2012). Studying the morphology and temporal evolution of these features can give insights into the dynamic atmospheric processes, as well as interactions between the planet's atmosphere and surface structures such as gravity waves, which may manifest themselves as visible wave trains at the cloud tops (Piccialli et al., 2014). This endeavor is crucial for understanding the energy transfer between different atmospheric layers and the global climate system of Venus.

Over the past decades, significant efforts at categorizing, classifying, and explaining the different types of cloud features found on Venus by carefully examining satellite images from missions such as the *Pioneer Venus Orbiter* (e.g. Rossow et al., 1980), *Venus Express* (e.g. Titov et al., 2012), and *Akatsuki* (e.g. Limaye et al., 2018; Peralta et al., 2019) have been made. However, manual classification is a labor-intensive process that is often constrained to specific regions or time frames and therefore limits the ability to extract comprehensive, large-scale insights from the full dataset. Moreover, manual classification is inherently subjective, which can lead to inconsistencies in the identification and categorization of cloud features, especially when working with large datasets from long-duration missions with images at various spatial scales, which all contain different types of features.

An automated, objective approach that can perform such a multi-scale classification is therefore essential for advancing our understanding of the Venusian atmosphere. This paper presents and investigates such an approach using unsupervised machine learning methods, which can leverage the entire datasets of *Venus Express*, *Akatsuki*, and future missions, as well as perform the classification task for features

* Corresponding author at: University Observatory Munich, Faculty of Physics, LMU Munich, Scheinerstr. 1, Munich, D-81679, Germany.
E-mail address: jmittend@usm.lmu.de (J. Mittendorf).

https://doi.org/10.1016/j.ascom.2024.100884
Received 4 July 2024; Accepted 1 October 2024
Available online 5 October 2024

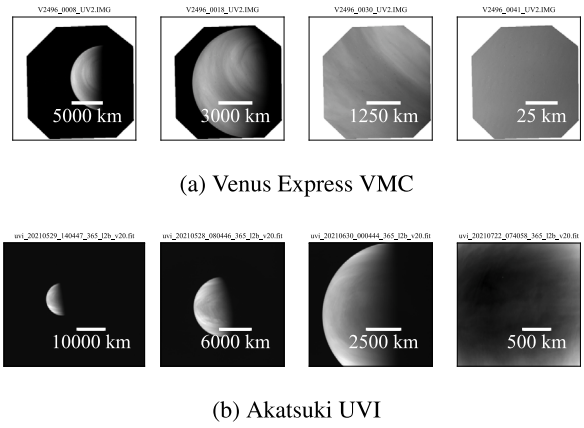


Fig. 1. Examples of raw satellite UV (365 nm) images for *Venus Express VMC* and *Akatsuki UVI*.

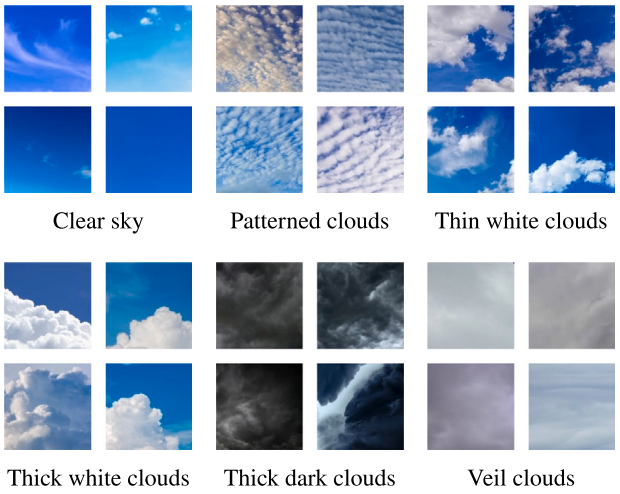


Fig. 2. Example images for each of the six *Cloud-ImVN 1.0* categories.

at arbitrary consistent scales where data with enough resolution is present. Since unsupervised methods do not require labeled data, they are particularly suited for planetary science, where such labels are generally not available and the uniqueness of extraterrestrial environments complicates traditional classification techniques. Partly adopting a method presented in a previous work on unsupervised classification of Earth cloud satellite imagery by Kurihana et al. (2019), we employ a deep convolutional neural network to extract high-level encoded feature vectors from a set of fixed-scale cloud patches which are then subsequently divided into classes using a clustering algorithm.

This paper is divided into the following major sections: Section 2 briefly introduces the datasets whose data will be ultimately used in the classification process. Section 3 explains the multi-step process for generating "virtual" cloud patches in nadir observation with minimal distortion from the raw satellite images, post-processing steps to improve the feature extraction results, as well as the complete classification framework. Section 4 shows and discusses the results of applying the unsupervised classification method to a benchmark dataset and multiple example Venus cloud patch datasets at different fixed scales. Finally, Section 5 discusses the implications of our findings for future studies of Venus and other planetary atmospheres.

2. Datasets

The datasets used in this paper consist of two Venus satellite imagery datasets (*Venus Express VMC* & *Akatsuki UVI*) and one benchmark dataset (*Cloud-ImVN 1.0*). Note that for convenience sake the files directly from the data archives will be referred to as *raw* data, even though they may have already been processed by the respective data providers.

2.1. Venus Express VMC

The first Venus satellite imagery dataset contains the images taken by the *Venus Monitoring Camera* (*VMC*) onboard *Venus Express* (*VEX*) (Markiewicz et al., 2007).[1] The spacecraft's orbit allows for studying small-scale cloud structures near Venus' North pole and complements the high quality larger scale and more global observations made by *Akatsuki* (see Section 2.2). In this paper we specifically focus on the 365 nm UV images as such are also available from the *Akatsuki* mission. The dataset consists of calibrated image files and corresponding geometry files which provide information about the incidence angle, emission angle, phase angle, latitude, and longitude for each pixel (Roatsch and Markiewicz, 2015). This information will later be used for generating the cloud patches (see Section 3.1). Fig. 1(a) shows four raw example images of this dataset.

2.2. Akatsuki UVI

The second Venus satellite imagery dataset comes from the *Ultraviolet Imager* (*UVI*) instrument of the *Akatsuki* spacecraft (Murakami et al., 2017, 2018; Yamazaki et al., 2018). Like for the *VEX VMC* dataset, we will only be focusing on the 365 nm UV images. Similarly, this dataset also contains calibrated image data files and geometry information files that map each pixel of an image to its corresponding latitude, longitude, local time, phase angle, incidence angle, emission angle, and azimuthal angle. Specifically, we use the calibrated *Level 2b* (*l2b*) *FITS* files in conjunction with the pointing corrected geometry *Level 3bx* (*l3bx*) *FITS* files.[2] Fig. 1(b) shows four raw example images of this dataset. It should be noted here, that while the bandwidths of the *VMC* (40 nm) and the *UVI* (14 nm) are different, the impact on the opacities and therefore also the morphological features should be minor. Furthermore, we will not be combining data from both datasets for any particular scale-classification in Section 4, so this difference is negligible for our results.

2.3. Cloud-ImVN 1.0

Cloud-ImVN 1.0 (Hoang, 2020) is a labeled Earth cloud image dataset that comprises 2100 images (150 × 150 px) equally distributed among six categories: clear sky, patterned clouds, thin white clouds, thick white clouds, thick dark clouds, and veil clouds (see Fig. 2 for examples).

With the exception of the additional "thin white clouds" category, the remaining five are identical to the ones found in the *SWIMCAT* (*Singapore Whole-sky Imaging Categories*) dataset (Dev et al., 2015), which contains a total of 784 images and of which this dataset is an extension. While this means that one could theoretically combine the images from both datasets, due to the equally distributed nature and the already significantly higher number of images in *Cloud-ImVN 1.0*, we only use the images from the latter. This dataset will act as a benchmark for testing the efficiency and accuracy of our classification method.

[1] Data available at: https://www.cosmos.esa.int/web/psa/venus-express.

[2] Data available at: https://darts.isas.jaxa.jp/planet/project/akatsuki/uvi.html.en.

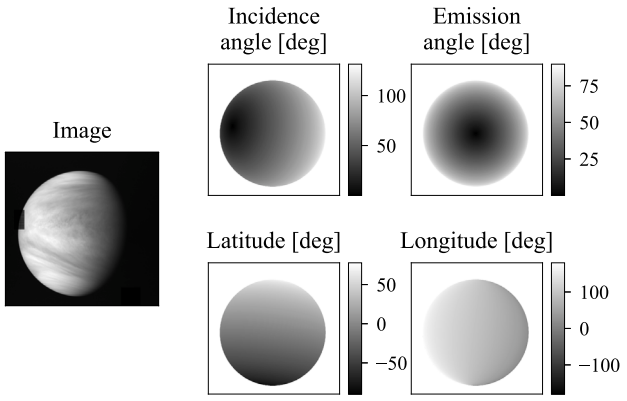


Fig. 3. Example of a raw *Akatsuki UVI* image with its corresponding geometry information.

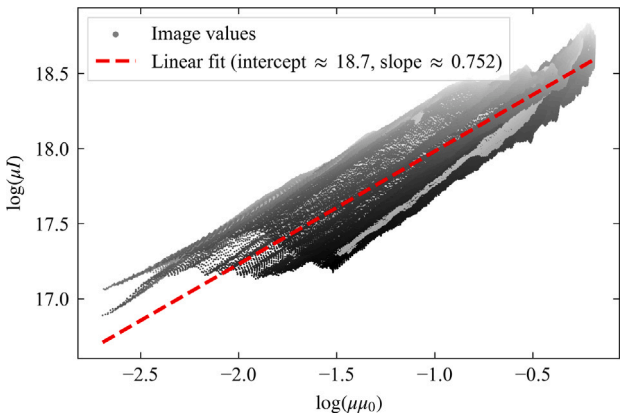


Fig. 4. Example scatter plot of the terms in Eq. (2) for an *Akatsuki UVI* image, including the linear fit whose slope yields the value of the limb-darkening coefficient $k \approx 0.752$. The brightness of the scattered points corresponds to the actual pixel values in the image.

3. Methods

3.1. Cloud patch generation

Cloud patches, *planet patches*, or simply *patches* for short are square images of a section of a planet's cloud deck or surface in nadir observation (i.e. viewed at a 90° angle to the surface) generated from satellite imagery via geometry information, projections, and post-processing steps. Collections of patches with a particular fixed physical scale will make up the patch image datasets for the subsequent classification task. The following subsections will explain the steps to generate such patches from the raw satellite datasets covered in Section 2.

As mentioned in Section 2, both Venus datasets contain geometry information files (see Fig. 3 for an example[3]) that can be used to easily map the image information back onto a three-dimensional sphere. We can therefore position a "virtual camera" around the planet to generate perfect nadir observation images from different viewpoints and interpolate the projected image points back onto a two-dimensional image.

3.1.1. Pre-processing steps

Prior to mapping the image data onto the 3D sphere, a number of pre-processing steps are conducted. The first step consists of removing all pixels marked as invalid, which automatically includes all empty space in the background, as it is not valid data in the geometry files. As a next step both the unilluminated parts of the planet (incidence angles >90°), as well as the parts close to a 90° observation angle (≡ emission angle) are removed based on a threshold angle value α. This is done to improve the results of the third pre-processing step, which is normalizing the image with respect to geometric brightness difference effects (i.e. illumination and observation angles) according to the Minneart law (Minnaert, 1941), following Titov et al. (2012):

$$I(\mu, \mu_0) = I_0 \times \mu^{k-1} \mu_0^k \,. \tag{1}$$

Here, I_0 is the theoretical intensity value of an image pixel at zero phase angle (i.e. in nadir observation and with the Sun directly behind the observer) and $I(\mu, \mu_0)$ is the actual measured intensity as a function of the cosine of the spacecraft zenith angle (≡ emission angle) μ and the cosine of the solar zenith angle (≡ incidence angle) μ_0 at that pixel. As reported by Titov et al. (2012), this law seems to generally be acceptable for normalization purposes on Venus when pixels with incidence and emission angles above 85° are excluded (i.e. for $\alpha \leq 85°$). Multiplying I with μ, one can derive from Eq. (1), that

$$\log(\mu I) = \log(I_0) + k \log(\mu \mu_0) \tag{2}$$

and therefore the limb-darkening coefficient k (Limaye, 1984) can be determined for an image via a linear fit through $\log(\mu I)$ vs. $\log(\mu \mu_0)$. Fig. 4 demonstrates this procedure on the example image. Once obtained, the intensity values for an image can then be normalized as

$$I_{\text{norm}} = \frac{I(\mu, \mu_0)}{\mu^{k-1} \mu_0^k} \,. \tag{3}$$

Fig. 5 illustrates all pre-processing steps mentioned thus far.

The final pre-processing step is the removal of any outlier pixels from the images. This is achieved by filtering the original image (after the first three pre-processing steps) with a median filter of kernel size 3×3 and masking any pixels as outliers whose absolute difference between the filtered version and the original is larger than $t_o \times \sigma_d$, where

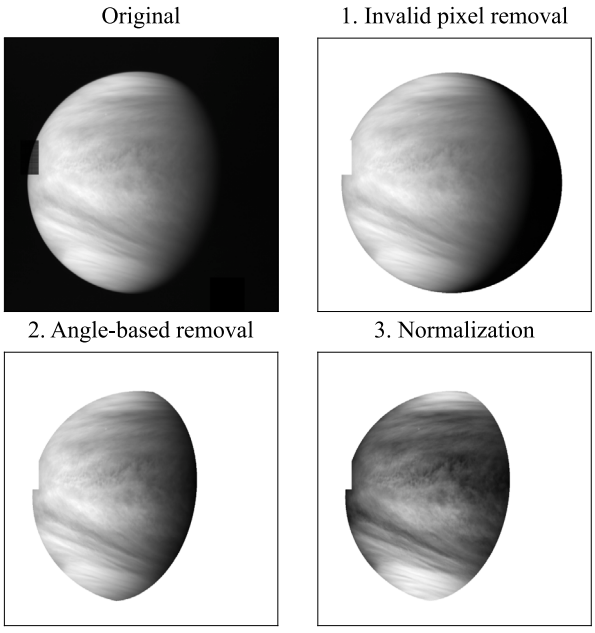


Fig. 5. The first three pre-processing steps applied to the raw satellite images, demonstrated for an *Akatsuki UVI* image with the threshold value for the angle-based removal step set to $\alpha = 75°$ (for both the incidence and emission angle).

[3] Unless otherwise noted, the raw satellite data files used for all example figures in this section are `uvi_20151207_052330_365_l2b_v10.fit` (image data) and `uvi_20151207_052330_365_l3bx_v10.fit` (geometry data).

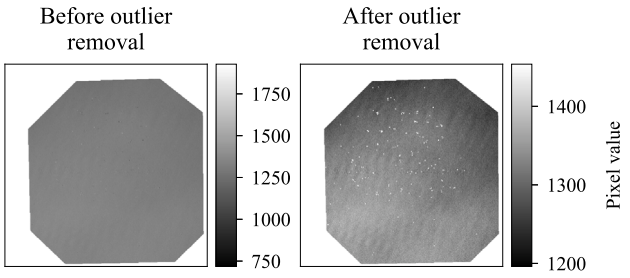


Fig. 6. Outlier removal pre-processing step illustrated on an example *VEX VMC* image with an outlier parameter threshold value of $t_o = 5$. (Original file: `V2496_0041_UV2.IMG`).

t_o is a free parameter and σ_d is the standard deviation of the difference image. The outlier pixels are then removed entirely from the data while the remaining gaps will be automatically filled at the interpolation step in Section 3.1.4. Fig. 6 shows an example of this outlier removal step applied to a *VEX VMC* image.

3.1.2. Spherical mapping and patch center pre-selection

Once a satellite image has been pre-processed, its corresponding longitude and latitude information are used to map the points back onto a three-dimensional sphere. To improve the spacing between the patches and to ensure that even close-up images of the planet that only produce small sections of the sphere on the scale of the patches themselves always have at least one virtual camera viewpoint in their center, the 3D sphere is rotated along the z- and y-axis so that the median vector of all points lies directly on the x-axis. The inverse of the required rotation matrix is stored so that the final patch center coordinates can be traced back to their corresponding original longitude, latitude, and local time. The virtual camera viewpoints (i.e. the patch centers) are generated as an equally spaced[4] grid on the sphere's surface in spherical coordinates with the angles $\Delta\phi_v$ and $\Delta\theta_v$ between viewpoints calculated from the respective patch scale s_p (e.g. 3000 km) of the dataset as

$$\Delta\phi_v = \Delta\theta_v = 2\arcsin\left(\frac{s_p}{2R_{pl}}\right), \tag{4}$$

where R_{pl} is the radius of the planet. A pre-selection of suitable patch centers is made by excluding those that lie outside the rectangular boundary of the actual data points. Fig. 7 shows an example of the viewpoints generated for the example satellite image and a patch scale of 3000 km.

3.1.3. Projections and density criteria

Once the viewpoints have been generated, the sphere of data points is repeatedly rotated along the z- and y-axis to center each viewpoint on $(1, 0, 0)$. In each instance the data points whose z- and y-coordinates are within 1.5 times half of the patch size in each direction are selected and their x-coordinates are discarded to create a flat projection along the x-axis. Fig. 8 shows two examples of such a projection for the example image.

To ensure that the number of data points in the projection roughly matches the expected patch resolution (i.e. to avoid strongly undersampled images after the final interpolation step), a simple density check is performed: If the number of points N_p is lower than $N_{p,min} = \rho_{p,min} \times r_p^2 \times 1.5^2$, where $\rho_{p,min}$ is an arbitrary minimum density parameter and

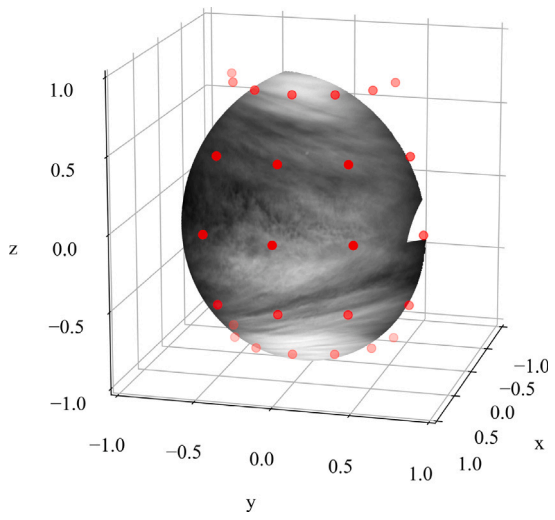


Fig. 7. Spherically mapped and shifted points of a pre-processed *Akatsuki UVI* image and pre-selection of patch centers/virtual camera viewpoints (red) for a patch scale of 3000 km.

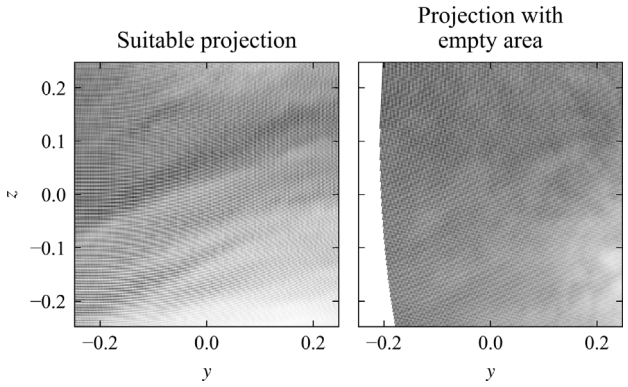


Fig. 8. Two examples for projections of data points for a 3000 km patch of an *Akatsuki UVI* image. The left projection's data points fully cover the patch area and are therefore suitable for the final interpolation step, while the right panel shows a projection with an empty area on its left side, which is not suitable for interpolation and is therefore ideally eliminated by the local density check (depending on the parameter settings).

r_p is the patch resolution in pixels, the projection is discarded.[5] If a projection has passed this "global" density threshold, another "local" density check is performed to ensure that there are no large empty areas in the projection (see the second panel of Fig. 8 for an example): The patch area is divided into $n_{bin} \times n_{bin}$ bins and the number of points in each bin N_{bin} is compared against the minimum required number of points in a bin $N_{bin,min} = \rho_{bin,min} \times (r_p/n_{bin})^2$, where $\rho_{bin,min}$ is another arbitrary minimum bin density parameter. If $N_{bin} < N_{bin,min}$ for any of the bins, the projection is also discarded.

3.1.4. Interpolation and patch images

Once a projection has passed both density checks, the last step towards the final patch image is the interpolation of its irregularly scattered points. To do so we use the *Python* library *scipy*'s (Virtanen et al., 2020) `scipy.interpolate.griddata` function with the

[4] The periodic nature of a sphere's surface makes it impossible to create a perfectly evenly spaced grid without an angle that is an integer ratio of 2π. However, since only one half of the planet can be visible at once, one can ensure that the non-evenly spaced points of the grid lie on the side without data points.

[5] The factor of 1.5 is related to the fact that the full projection area is 1.5 times the size of the final patch image in both x- and y-direction and should therefore give the density parameter $\rho_{p,min}$ a more intuitive relation to the pixel density in the final image.

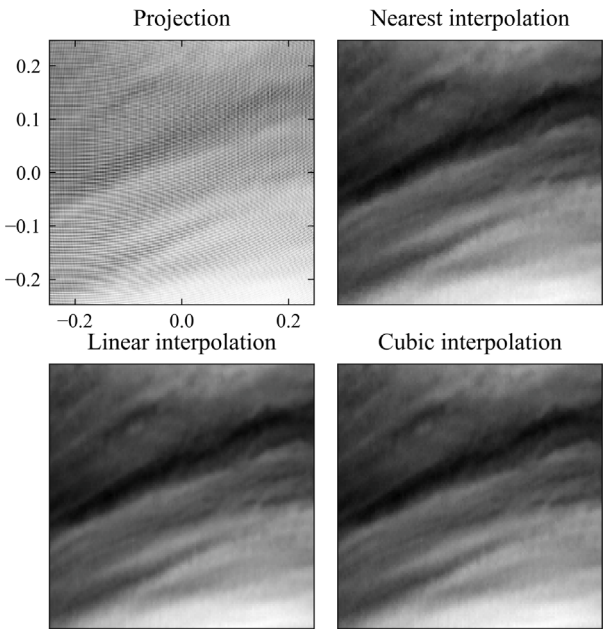


Fig. 9. Example of the final interpolation step for an *Akatsuki UVI* patch image of 3000 km scale and 128×128 resolution, as well as a comparison between three different interpolation methods: nearest (neighbor), linear, and cubic. One can clearly see that the difference between the nearest neighbor interpolation and the linear/cubic interpolations is only minute.

nearest interpolation method.[6] Since the density checks above should ensure that the number of points in the projection roughly matches the patch image resolution, a nearest neighbor interpolation provides sufficient results while also being much faster than a higher quality linear or cubic interpolation. Fig. 9 demonstrates the interpolation step for an example patch, as well as comparisons between different interpolation methods.

These steps are applied to all images in a satellite dataset and create the full patch datasets used for the classification task.

3.1.5. Patch flattening and standardization

In order to help the machine learning tools focus more on small-scale details and features rather than large-scale brightness gradients and differences, an additional but theoretically optional flattening + standardization step is applied to all images of a completed patch dataset. The flattening procedure consists of subtracting a Gaussian-blurred version of each patch from itself, where the kernel size k_f and standard deviation σ_f used for the Gaussian blurring are again free parameters and determine the strength of the flattening procedure. Standardization is then applied by subtracting the mean of each image from itself, dividing it by its standard deviation, multiplying it with a contrast parameter c_s, and finally clipping its values to a range of $[-1, 1]$. Fig. 10 shows the flattening procedure applied to an example patch.

3.2. Encoding the patches

Once a full patch dataset has been generated, the first step towards the classification is to encode each image into its latent representation via a deep convolutional neural network (CNN). The resulting feature vectors will then be clustered using a clustering algorithm (see Section 3.3), following the methodology of Kurihana et al. (2019).

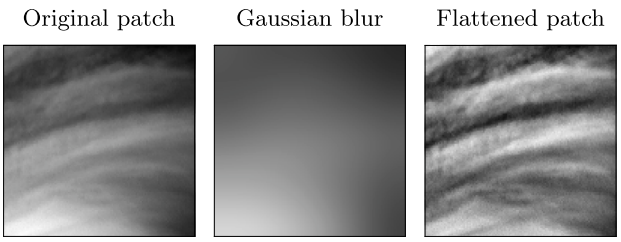


Fig. 10. Example of the patch flattening and standardization process applied to a 3000 km *Akatsuki UVI* patch of resolution 128×128. The standard deviation of the Gaussian blur was set to $\sigma_\mathrm{f} = 16$, which corresponds to a physical size of 375 km, the blurring kernel size was set to $k_\mathrm{f} = 2 \times 3\sigma_\mathrm{f} + 1 = 97$, and the standardization contrast was set to $c_\mathrm{s} = 0.33$.

3.2.1. Choice of encoder base model

Because Chen et al. (2020) propose to use a *ResNet* (He et al., 2016) base encoder for training with *SimCLR* (*A Simple Framework for Contrastive Learning of Visual Representations*, more details in Section 3.2.2), we choose to use a *ResNet-18* model as our base model since its architecture is comparable to the encoder used by Kurihana et al. (2019). Additionally, as our patch datasets are quite small (1708–10 734 images, see Section 4) and a default *ResNet-18* already has over 11 million trainable parameters, larger models would only increase the chance of overfitting and unnecessarily raise the required training time. The base model is initialized with weights obtained from pre-training on the *ImageNet-1K* dataset (Russakovsky et al., 2015).[7] Because the final layer of this model represents a probability distribution over the 1000 categories in the *ImageNet-1K* dataset and our data does not fall under these categories, it is removed from the base model. Hence, the second-to-last layer of the base model acts as the feature vector of length 512 for each input image. Fig. 11 shows an illustration of the resulting architecture.

3.2.2. Choice of training method

The results of testing a selection of model architectures and training methods shown in Table 1 reveal that even a simple encoder consisting only of the pre-trained base model without any actual training on our own data already has sufficient feature recognition and extraction capabilities to provide high clustering accuracies on the benchmark dataset and is better than explicitly training a simple *Autoencoder* (with a decoder that mirrors the base model) on our dataset such as Kurihana et al. (2019). The latter is likely limited by the low amount of available training data in our case (only 2100 images in the benchmark dataset) and cannot learn good latent representations purely by learning reconstructions of the input images, even when applying augmentations (i.e. transformations such as rotation, horizontal flipping etc.) to the images. We also found that even after the patch flattening step, the *Autoencoder*'s embeddings were highly sensitive to large-scale gradients and artifacts such as holes, which even with the precautions taken in step 3.1.3 can still occasionally occur. However, with the right combination of additional dimensionality reduction and clustering methods (see Section 3.3), a model trained on the benchmark images using *SimCLR* by Chen et al. (2020) provides the best results (see also Section 4.1). For training a model with *SimCLR*, the base model is appended with a small multilayer perceptron (MLP) *projection head* with one hidden layer that projects the feature vector into a lower dimensional space where a *contrastive loss* is calculated. In our case this projection head consists of two fully connected layers with respective output sizes of 256 and 64 and a *ReLU* activation function after the first.

[6] https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.griddata.html.

[7] https://pytorch.org/vision/main/models/generated/torchvision.models.resnet18.html.

Table 1
Mean total accuracies obtained from classifying the benchmark *Cloud-ImVN 1.0* dataset with various combinations of encoder models, additional reduction methods, reduction dimensions, and clustering methods. To determine the mean values, each combination of methods was ran for a total of eight times. The *Simple* model refers to a simple modified *ResNet-18* model pre-trained on the *ImageNet-1K* dataset as explained in Section 3.2.2, while the *SimCLR* and *Autoencoder* models were trained specifically on the benchmark dataset. *PCA* is short for *Principle Component Analysis* and *HAC* refers to *Hierarchical Agglomerative Clustering*.

Encoder model	Red. method	Red. dims.	Clustering method	Mean tot. accuracy [%]	Std. deviation [%]
SimCLR	t-SNE	3	k-Means	84.9(19)	5.3
SimCLR	t-SNE	3	HAC	84.8(4)	1.0
SimCLR	None	None	HAC	84.5	0.0
Simple	t-SNE	3	k-Means	82(3)	6
Simple	t-SNE	3	HAC	81.7(18)	4.9
SimCLR	PCA	256	HAC	81.4(11)	2.9
SimCLR	PCA	64	HAC	76.9(5)	1.4
SimCLR	PCA	64	k-Means	75.1(3)	0.6
SimCLR	None	None	k-Means	74.7(3)	0.6
SimCLR	PCA	16	HAC	74.4	0.0
SimCLR	PCA	256	k-Means	74.2(7)	1.7
Simple	None	None	HAC	73.0	0.0
SimCLR	PCA	16	k-Means	72.4(15)	4.2
Simple	PCA	256	k-Means	72.3(6)	1.7
Simple	PCA	64	k-Means	72.3(8)	2.1
Simple	PCA	64	HAC	72.3(10)	2.6
Simple	None	None	k-Means	70.1(8)	2.0
Simple	PCA	256	HAC	69.2(14)	3.8
Simple	PCA	16	k-Means	68(3)	7
Simple	PCA	16	HAC	64.24(9)	0.24
Autoencoder	PCA	16	HAC	63.8	0.0
Autoencoder	t-SNE	3	k-Means	62.0(14)	3.9
Autoencoder	PCA	64	HAC	61.2(11)	2.9
Autoencoder	None	None	HAC	59.9	0.0
Autoencoder	PCA	256	HAC	59.9	0.0
Autoencoder	t-SNE	3	HAC	59.3(17)	4.8
Autoencoder	None	None	k-Means	57.2(12)	3.2
Autoencoder	PCA	16	k-Means	56.3(10)	2.7
Autoencoder	PCA	64	k-Means	56.2(15)	4.1
Autoencoder	PCA	256	k-Means	54.4(13)	3.6

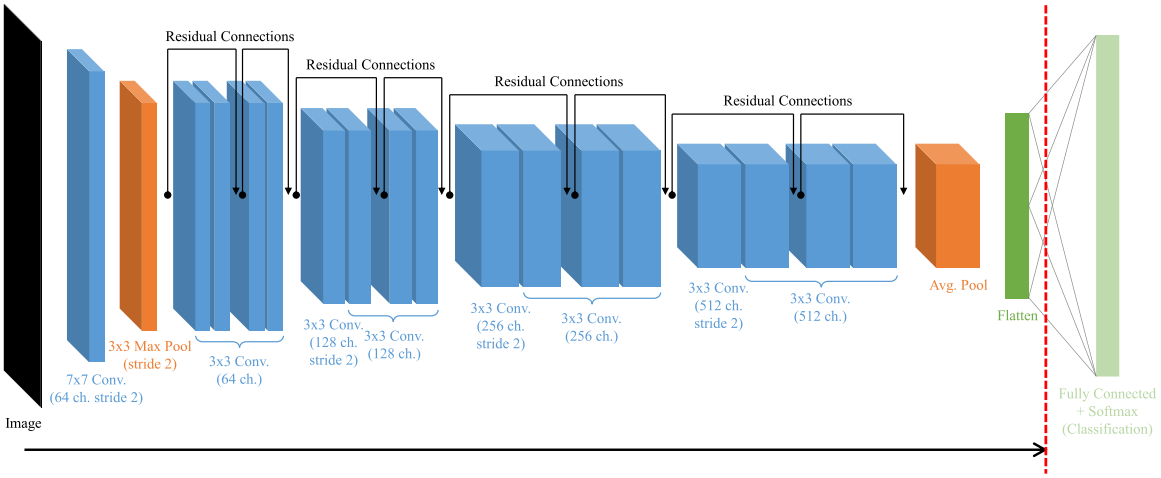


Fig. 11. Architecture of the modified *ResNet-18* base model used for encoding the patches. The data flows from the original patch image on the left to the *Flatten* layer on the right, where the final feature vector is obtained. The fully connected layer that is part of the original model is removed from our version.

The main idea behind *SimCLR* is to train a model to maximize the agreement between the model outputs of two augmented versions of the same input image. Specifically, for a given batch of size N, each image in the batch is augmented twice to form a set of $2N$ augmented images. Each pair of such augmented images that originates from the same original image is defined as a positive pair, while all other $2(N-1)$ images are treated as negative examples, respectively. One can then define the *NT-Xent* (normalized temperature-scaled cross entropy) loss function for a positive pair (i, j) as

$$l_{i,j} = -\log \frac{\exp\left(\mathrm{sim}\left(z_i, z_j\right)/\tau\right)}{\sum_{k=1}^{2N} \mathbb{1}_{[k\neq i]} \exp\left(\mathrm{sim}\left(z_i, z_k\right)/\tau\right)}, \tag{5}$$

where z_i is the output of the model for an input $\tilde{x}_i$, $\mathrm{sim}\left(z_i, z_j\right) = z_i^{\top} z_j / \left(\|z_i\| \, \|z_j\|\right)$ is the cosine similarity between two output vectors, $\mathbb{1}_{[k\neq i]}$ is equal to 1 if $k \neq i$ and 0 otherwise, and τ is a temperature parameter. For each batch, the sum of this loss function applied to all

positive pairs (i, j) and (j, i) in the batch (normalized via a division by $2N$) is then the full loss function to be minimized during training. The augmentations we employ are a random rotation (cropped to eliminate empty areas) of the input image, a random resized crop with a scale range of $[0.5, 1]$, a random horizontal flip, color jitter[8] with the brightness and contrast parameters set to 0.5 and no hue/saturation jitter, and a random application of Gaussian blur with a probability of 50%, a kernel size of 25 and a blurring standard deviation range of $[0.1, 2]$. These augmentations help the model to learn rotationally invariant embeddings that are also less sensitive to changes in resolution and total image brightness and contrast, which is ideal for our Venus cloud patches. Just like Chen et al. (2020), we also use the *LARS* (*Layer-wise Adaptive Rate Scaling*) optimizer (You et al., 2017, preprint) to perform the gradient descent steps, but without learning rate warm-up or decay.

3.2.3. Normalization

During pre-training of the base model on the *ImageNet-1K* dataset, normalization was applied to all input images so that the mean value of each channel over the entire dataset is 0 and the corresponding standard deviation is 1. To effectively use our own data with the model, the same transformation must be applied to our images. This can be achieved by simply subtracting the means $\epsilon_{\mathrm{RGB}} = (0.485, 0.456, 0.406)$ of the *ImageNet-1K* channels from the respective patch image channels and then dividing the resulting values by the corresponding standard deviations $\sigma_{\mathrm{RGB}} = (0.229, 0.224, 0.225)$.[9] However, since all Venus patch images are single-channel grayscale images and the normalization values for each RGB channel are slightly different, this transformation would introduce colors into the images. For single-channel patch images, we therefore only use a single mean $\epsilon_{\mathrm{gs}} = 0.4589225$ and standard deviation $\sigma_{\mathrm{gs}} = 0.2255861$ value for normalizing all three RGB channels, both of which are calculated from the full normalization RGB vectors via the color to luma (L) conversion formula used by *PyTorch*[10]:

$$L = 0.2989\,R + 0.587\,G + 0.114\,B \tag{6}$$

Additionally, all images are resized to 224×224 before being passed through the model, as that is the native input resolution.[11]

3.3. Clustering

Once a full patch dataset has been encoded with a model previously trained on the patches, they are finally divided into classes by clustering the base model's output feature vectors for each image with a clustering algorithm, following the methodology of Kurihana et al. (2019). As Table 1 shows, we also tested using an additional dimensionality reduction step between the encoding and clustering step with *Principal Component Analysis* (*PCA*) and *t-SNE* (*t-Distributed Stochastic Neighbor Embedding*) (van der Maaten and Hinton, 2008) on the benchmark dataset. The latter is a statistical algorithm that maps high-dimensional data into three- or two-dimensional space in such a way that similar points in the original domain maintain proximity in the output map with a high probability. While we are using it as an intermediate dimensionality reduction step here, it is also particularly useful for visualizing high-dimensional clusters and inspecting how well the model assigns similar encodings to images belonging to the same class, as will become apparent in Section 4.1. On average, clusters obtained with *k-means* clustering after an additional three-dimensional *t-SNE* reduction are the most accurate for both a *Simple* encoder and an encoder trained with *SimCLR*. This is likely due to the *t-SNE* algorithm producing better separated "islands" (see e.g. the 2D examples in Fig. 12) and *k-means* clustering generally performing better on lower-dimensional data. However, not only can the *t-SNE* algorithm sometimes introduce islands that are not "real" clusters, but these accuracies are also highly variant accross different iterations of the clustering pipeline, as both *t-SNE* and *k-means* depend on random initialization. We therefore choose to use the option with no additional reduction method and *hierarchical agglomerative clustering* (*HAC*) with *Ward*'s method (Ward, 1963), as the corresponding results are much more stable for different iterations of the clustering approach and the accuracy is only negligibly smaller than the highest score for the *SimCLR* encoder (see also Section 4.1). Unlike *k-means* clustering, *HAC* also has the advantage of not assuming equally sized clusters, which we cannot necessarily expect for our Venus cloud patch classes.

Table 2
Parameters used for training the encoder model with *SimCLR* and the *LARS* optimizer.

Parameter	Value
Batch size	32
Max. epochs	64
Loss temperature	1
Base learning rate	0.0375
LARS coefficient	0.001
LARS momentum	0.9
LARS weight decay	0.0001

All neural network related tasks are performed using the *PyTorch* library (Paszke et al., 2019) for *Python*, while the *scikit-learn* library (Pedregosa et al., 2011) is employed for performing the clustering and dimensionality reduction steps (including all *t-SNE* embeddings). Unless indicated otherwise, we use the default values for all corresponding functions and classes.

4. Results

4.1. Benchmark on Cloud-ImVN 1.0

To further validate the choice of methods outlined in Sections 3.2 & 3.3, we will first evaluate their performance on the *Cloud-ImVN 1.0* benchmark dataset. Because our Venus cloud patches are grayscale images, both the original full color version, as well as a grayscale version (using Eq. (6)) of the benchmark dataset are tested. The number of classes for the clustering algorithm is set to 6. The parameters for training the encoder model (which are also used for the Venus patch results in Section 4.2) can be found in Table 2. The datasets are split into a training and test dataset of sizes 80% and 20%, respectively. Once a model has been trained for the maximum number of epochs, the model weights after the epoch with the lowest loss on the test dataset are chosen as the final weights to minimize overfitting.

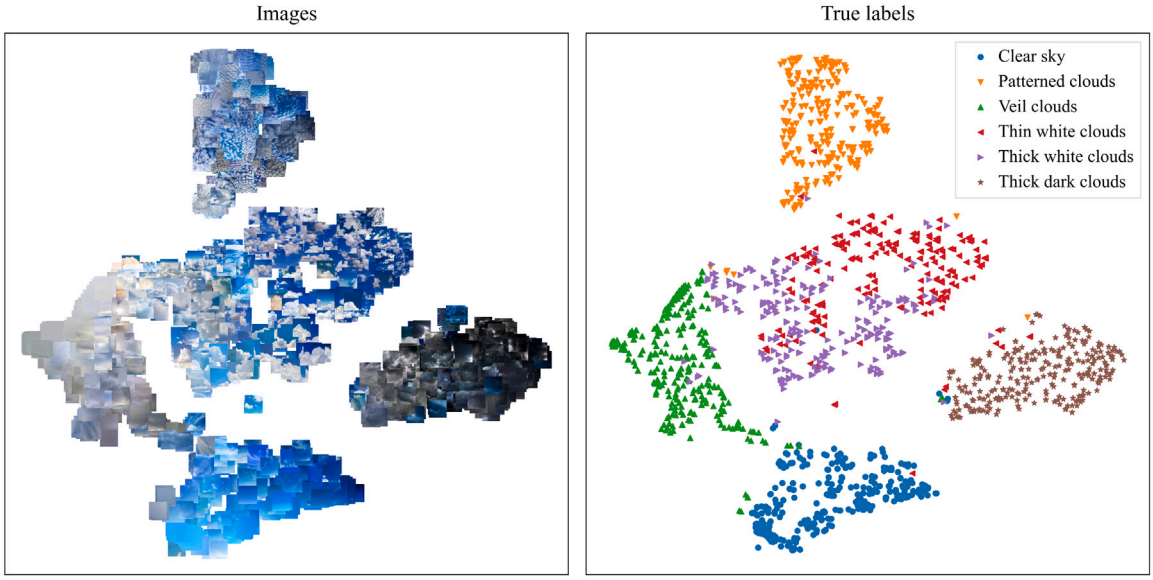
Fig. 12 shows a *t-SNE* embedding of the encoded image feature vectors, including the actual images and their corresponding class labels. It is already apparent here that images belonging to the same class are also assigned similar feature vectors, as ideally expected. Additionally, except for the "thin white clouds" and "thick white clouds" category, which are often visually similar even for a human observer, as well as the "clear sky" and "veil clouds" category, which due to their often flat and almost featureless nature are very difficult to distinguish in the grayscale version, the classes seem to fall into distinct clusters, further supporting the clustering approach for classification. It should be noted that a failure to distinguish between "clear sky" and "veil clouds" images does not raise a problem in the case of Venus as clouds are omnipresent and the former category is therefore non-existent.

[8] https://pytorch.org/vision/main/generated/torchvision.transforms.ColorJitter.html.

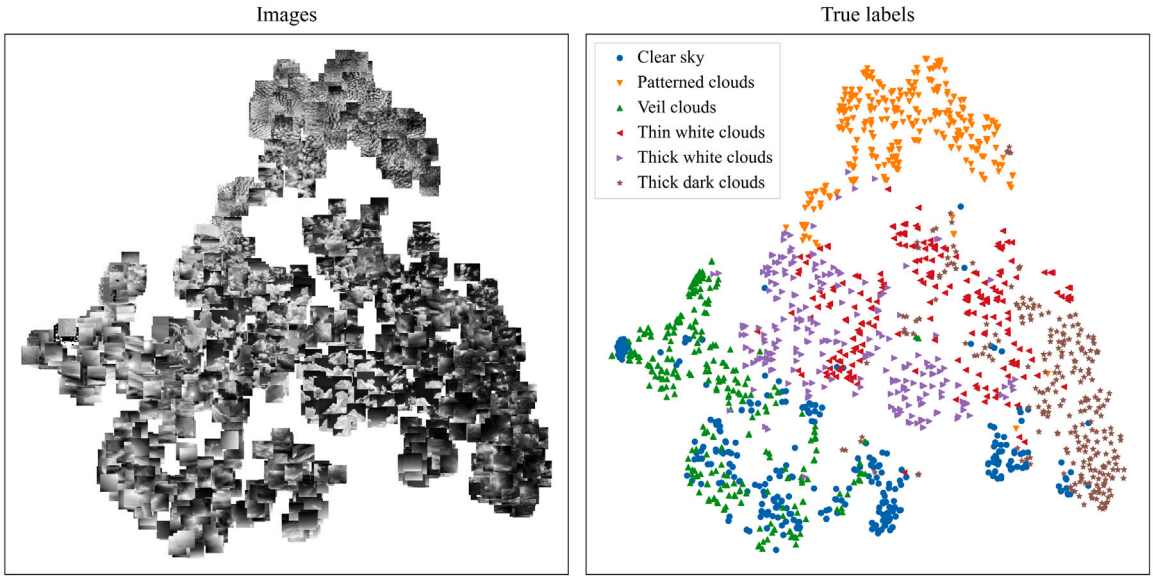
[9] https://pytorch.org/vision/main/models/generated/torchvision.models.resnet18.html#torchvision.models.ResNet18_Weights.

[10] https://github.com/pytorch/vision/blob/main/torchvision/transforms/_functional_tensor.py#L146.

[11] These normalization steps are part of the encoding section because the transformations are applied to the dataset images at encoding time in our code, however, one could also choose to apply them as a permanent last step in the patch dataset generation process.



(a) Full color dataset



(b) Grayscale dataset

Fig. 12. *t-SNE* embedding of the encoded *Cloud-ImVN 1.0* dataset for both the full color and grayscale version. The left panels show the actual images and the right panels show the corresponding true class labels.

To allow for the calculation of an accuracy score, the predicted class labels are matched to the true labels by finding the mapping that maximizes the total accuracy. The latter is calculated from the average of the individual class accuracies (the number of members is equal for each class, so no weighting is necessary), which are the ratio of correctly predicted labels to total class members.

Fig. 13 and Table 3 show the result of applying our classification approach to the dataset. For the full color dataset, the classification achieves a total accuracy of 84.5% with a high agreement between the true and predicted class labels. The largest discrepancy is found for predicting the labels of members of the "thin white clouds" category which are sometimes assigned to the "thick white clouds" category instead. Some images from the "veil clouds" category are also classified as "thick white clouds". A closer examination of the *t-SNE* mappings in Figs. 12(a) and 13(a) reveals that most of these misclassification occur for images in a "transition zone" between the two categories, where clouds from both categories are blended. As will be shown in Section 4.2, such cases also present a challenge for our classification method with the Venus cloud imagery. In the case of the grayscale dataset a lower accuracy of 59.7% is achieved with most inaccuracies occurring between the "thick dark clouds" and "thin white clouds" category, the "veil clouds" and "clear sky" category, which as already explained above is to be expected, and the "thin white clouds" and "thick white clouds" category, similar the full color version. As expected from the loss of color information, the total accuracy is lower in the grayscale case, but still significantly higher than the baseline accuracy of 1/6 for completely random labels.

4.2. Venus patch classification

The previous section showed that our unsupervised classification method is able to achieve good accuracies when applied to the benchmark dataset. We will now present the results on three example Venus

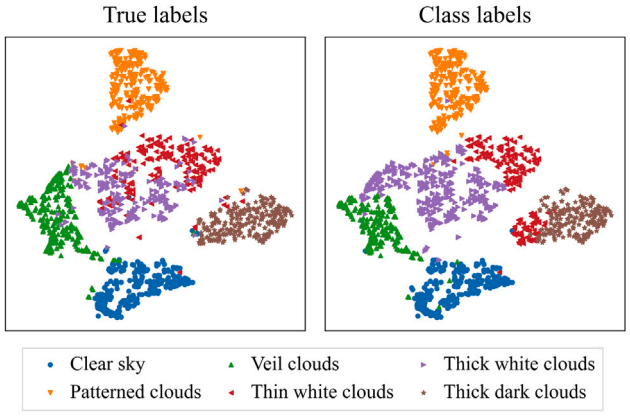
Table 3
Confusion matrices (predicted vs. true classes) and class/total accuracies for the example *Cloud-ImVN 1.0* classification results in Fig. 13.

(a) Full color dataset

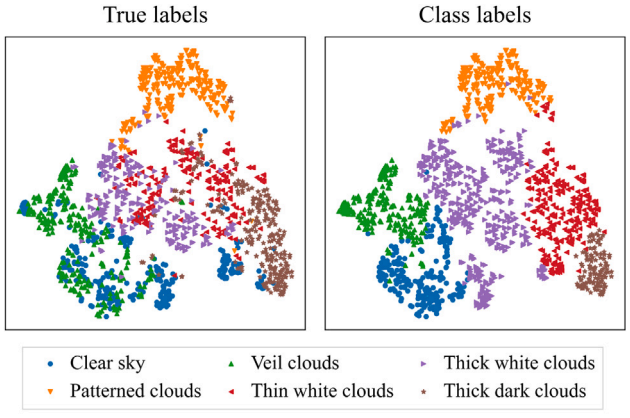
True classes	Predicted classes						Accuracy [%]
	Clear sky	Patterned	Veil	Thin white	Thick white	Thick dark	
Clear sky	339	0	4	3	4	0	96.9
Patterned	0	339	0	2	7	2	96.9
Veil	8	0	260	3	79	0	74.3
Thin white	0	4	0	229	108	9	65.4
Thick white	0	3	6	12	327	2	93.4
Thick dark	0	0	0	69	0	281	80.3
Total							84.5

(b) Grayscale dataset

True classes	Predicted classes						Accuracy [%]
	Clear sky	Patterned	Veil	Thin white	Thick white	Thick dark	
Clear sky	157	0	41	44	94	14	44.9
Patterned	0	308	0	20	22	0	88.0
Veil	123	0	215	0	12	0	61.4
Thin white	2	2	1	154	191	0	44.0
Thick white	20	1	44	11	274	0	78.3
Thick dark	0	5	0	152	47	146	41.7
Total							59.7



(a) Full color dataset (total accuracy: 84.5 %)



(b) Grayscale dataset (total accuracy: 59.7 %)

Fig. 13. Unsupervised classification approach applied to the full color and grayscale version of the *Cloud-ImVN 1.0* dataset, visualized with a *t-SNE* embedding of the feature vectors (also see Fig. 12). The right panels show the resulting labels of the classification which have been matched to the true labels in the left panels. Table 3 contains the corresponding confusion matrices and individual class accuracies.

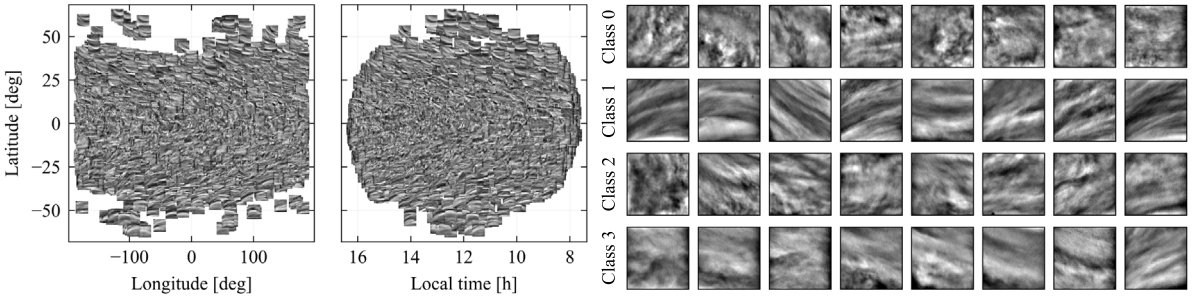
Table 4
Parameter values (see Section 3.1) used for generating the Venus patch datasets in Section 4.2.

Parameter	Value
Angle-based removal threshold	$\alpha = 75°$
Outlier removal sigma multiplier	$t_\sigma = 5$
Minimum patch density	$\rho_{\mathrm{p,min}} = 0.5$
Local density bins	$n_{\mathrm{bin}} = 8$
Minimum bin density	$\rho_{\mathrm{bin,min}} = 0.1$
Interpolation method	Nearest
Resolution	$r_{\mathrm{p}} = 128$
Flattening blur sigma	$\sigma_{\mathrm{f}} = 16$
Flattening blur kernel size	$k_{\mathrm{f}} = 97$
Standardization contrast	$c_{\mathrm{s}} = 0.33$

patch datasets at different scales. A 3000 km and 500 km dataset generated from the *Akatsuki UVI* 365 nm satellite images provide cloud patches on a global and medium scale with high contrast features, and a 25 km dataset generated from the *Venus Express* UV images provides close-up patches on the scale of some of the wave features discussed by Piccialli et al. (2014). In all cases the number of clusters/classes is set to 4, as higher numbers tend to be difficult to interpret. All necessary parameter values that were used for generating the patch datasets as described in Section 3.1 are listed in Table 4.

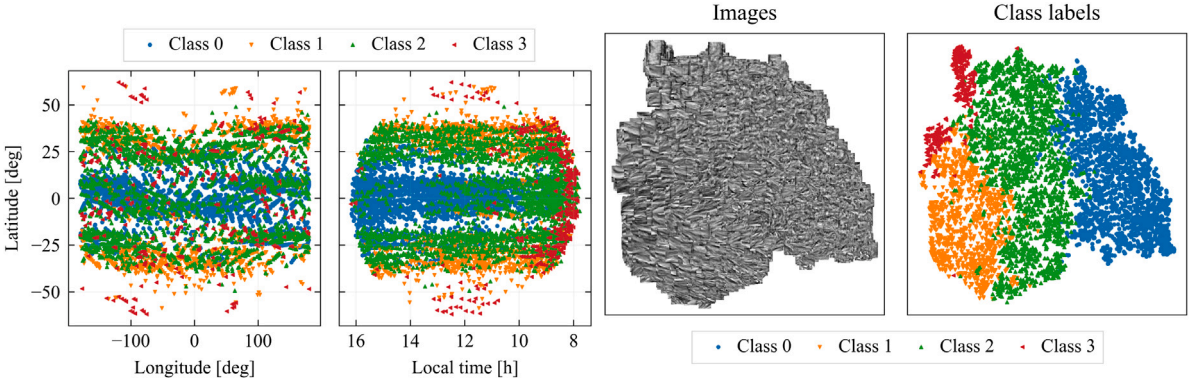
4.2.1. 3000 km patches (Akatsuki UVI 365 nm)

The 3000 km dataset contains global-scale cloud patches in a range from −60° to 60° in latitude and 8 h to 16 h in local time with full longitudinal coverage (see Fig. 14(a)) and consists of a total of 6474 patches. The results of the 4-class classification pipeline can be seen in Figs. 14(b)–14(d), which show a random selection of images from each cluster, the class labels on the same latitude vs. longitude / local time scatter plot as in Fig. 14(a), and the class labels on a *t-SNE* visualization of the feature vectors like in the case of the benchmark dataset. Looking at Fig. 14(c), one can see that the classes clearly capture different distinct regions in latitude vs. local time. Class 0 consists mostly of the turbulent and patterned clouds in the afternoon convective wake, while class 1 contains the streaky clouds found in mid- to high-latitude regions (Titov et al., 2012; Rossow et al., 1980). Class 3 seems to mostly capture clouds near the morning edge of the local time distribution, which could potentially be linked to a small gradient artifact on the right side of many of such patches (see Fig. 14(b)) but also to the morphological features which are generally a mix of streaky and patchy



(a) Central latitudes vs. longitudes and local times of the 3000 km Venus cloud patches.

(b) Random examples for each of the four classes of the 3000 km Venus cloud patches.



(c) Central latitude vs. longitude and local time distribution of the 3000 km Venus cloud patch classes.

(d) Real images and corresponding classes of the 3000 km Venus cloud patches visualized on a *t-SNE* embedding of the feature vectors.

Fig. 14. Results of the classification framework applied to the 3000 km Venus cloud patches generated from the *Akatsuki UVI* 365 nm images.

patterns and less pronounced than the previous two classes. Class 2 is harder to interpret and appears to simply be the rest of the patches that do not belong to any of the other three classes.

Comparing the *t-SNE* embedding of the feature vectors in Fig. 14(d) to the *t-SNE* embedding of the *Cloud-ImVN 1.0* dataset in Fig. 12, one can see that there is a clear difference in their distribution patterns. Unlike in the latter case, where discrete clusters are evident, the embeddings in the former exhibit a much more uniform distribution without clear boundaries between classes. This observation is crucial, as it highlights an intrinsic challenge in classifying cloud formations on planets with extensive cloud coverage like Venus. The cloud patterns captured in our patches do not necessarily always correspond to isolated categories but are part of a continuous distribution of features that cover the entire planet. This inherently complicates the classification task, as features often blend into one another and a particular patch may therefore not always be assigned to a distinct type of cloud. This limitation becomes increasingly apparent as we go to the medium scale 500 km patches in the next section, where the resolution of distinctive features decreases, further blurring the lines between categories and challenging our approach of classifying unique cloud formations.

4.2.2. 500 km patches (Akatsuki UVI 365 nm)

The 500 km patch dataset consists of 1708 images and covers a similar range as the 3000 km patches but extends slightly further South to −70° latitudes (see Fig. 15(a)).[12] Compared to the 3000 km patches, structures at this scale are more difficult to distinguish from one another, with most patches showing some form of patterned features of varying contrast or almost no features at all. This is also visible in the classes, which – as can be seen on Fig. 15(b) – are less distinct from another than in the previous example. While all classes seem to contain images with patterned clouds, class 1 and 3 mostly consist of images with higher contrast features in the afternoon region (see Fig. 15(c)), whereas class 0 holds images that are more flat and almost featureless, but also occasionally show slightly more streaky features. Similar to the previous case, class 2 seems to simply carry the images that do not belong to any of the other classes. The *t-SNE* visualization of the feature vectors in Fig. 15(d) again shows no real cluster separation between the different classes, complicating a separation into distinct categories.

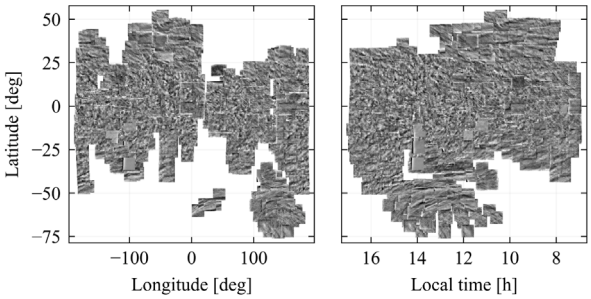
4.2.3. 25 km patches (VEX VMC UV)

The small-scale 25 km patch dataset contains 10 734 images and has a much smaller latitudinal extent than the previous two datasets due to the orbit of *Venus Express* (see Fig. 16(a)). At this scale the large majority of images is close to featureless with only a few images showing the expected wave patterns. This is also apparent in the classes which, as can be seen in Fig. 16(b), consist of three classes that only contain almost featureless images and class 3, which holds most of the patches with visible waves. However, the latter also contains multiple images without such features, which can again be attributed to the fact, that the wave patches are continuously blended into the rest of the images (see Fig. 16(d)) and therefore no clear boundary can be drawn between them and the other patches.

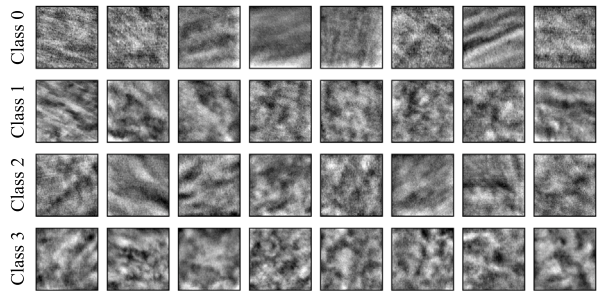
5. Conclusions

In this study we introduced a novel framework to generate and classify nadir observation patches of Venus' clouds at various consistent

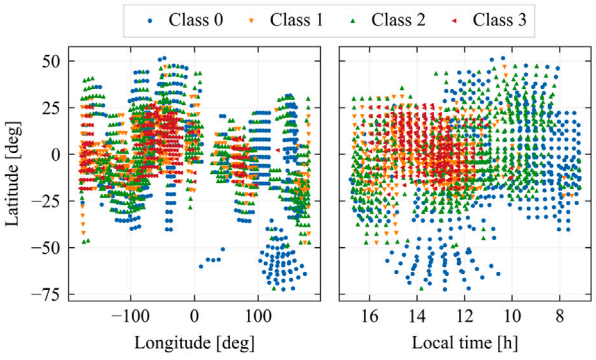
[12] While a much larger amount (20 000+) of patches at this scale could be generated from the *VMC* data, we found that a significant portion of these patches are either very noisy or contain visual artifacts from the instrument and would therefore require manual cleanup, which is contrary to our idea of finding an automated framework.



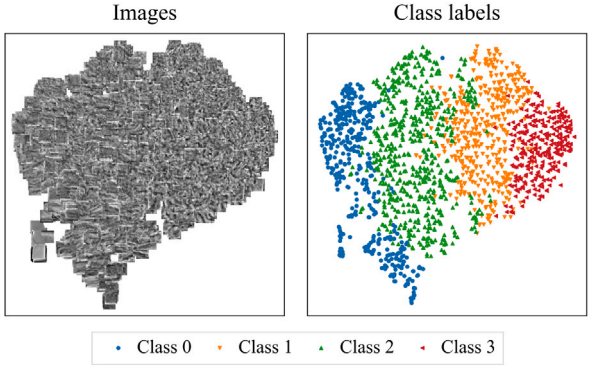
(a) Central latitudes vs. longitudes and local times of the 500 km Venus cloud patches.



(b) Random examples for each of the four classes of the 500 km Venus cloud patches.

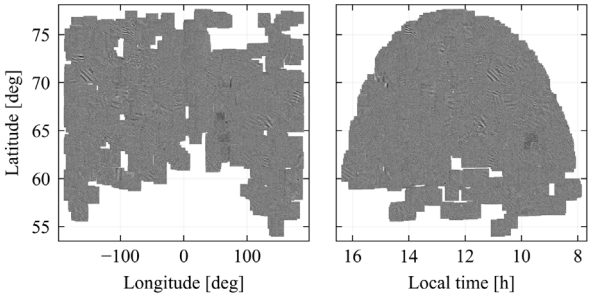


(c) Central latitude vs. longitude and local time distribution of the 500 km Venus cloud patch classes.

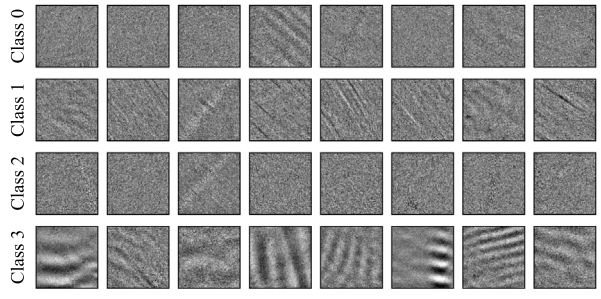


(d) Real images and corresponding classes of the 500 km Venus cloud patches visualized on a *t-SNE* embedding of the feature vectors.

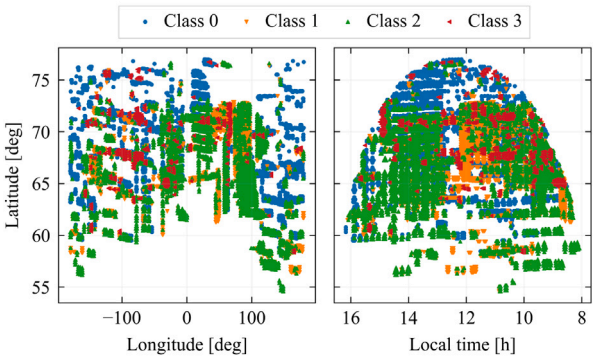
Fig. 15. Results of the classification framework applied to the 500 km Venus cloud patches generated from the *Akatsuki UVI* 365 nm images.



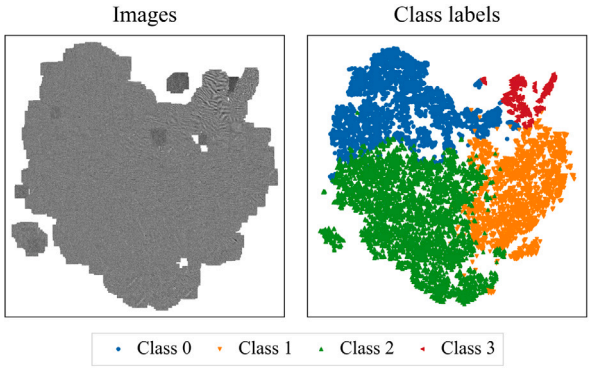
(a) Central latitudes vs. longitudes and local times of the 25 km Venus cloud patches.



(b) Random examples for each of the four classes of the 25 km Venus cloud patches.



(c) Central latitude vs. longitude and local time distribution of the 25 km Venus cloud patch classes.



(d) Real images and corresponding classes of the 25 km Venus cloud patches visualized on a *t-SNE* embedding of the feature vectors.

Fig. 16. Results of the classification framework applied to the 25 km Venus cloud patches generated from the *Venus Express* UV images.

scales through the use of unsupervised machine learning techniques. We found that our approach is able to achieve promising accuracy when applied to a curated benchmark dataset of Earth cloud categories and is able to identify meaningful classes for global-scale cloud features on Venus, as well as capture the wave patterns often found at small scales. However, at medium scales we encountered challenges as resolution and distinctive features start to diminish and blended features complicate the separation of well defined clusters.

Experiments with advanced unsupervised classification methods such as replacing the clustering stage with *SCAN*, a full unsupervised deep learning framework presented by Van Gansbeke et al. (2020), were not able to improve upon the results achieved with our presented method. A possible explanation for this may be, that in the context of machine learning tools, which generally depend on extensive amounts of available training data, our patch datasets are still relatively small, thereby constraining the learning capacity of the neural networks. This also represents a limitation of our method, as it currently relies on a base model CNN that has been pre-trained on a large number of unrelated images to enhance its general image recognition capability. Hybrid approaches that combine both supervised and unsupervised methods may offer a solution by incorporating additional information to improve feature separation and classification accuracy.

Overall, these findings represent a significant advancement, as this automated framework offers a scalable and objective solution for analyzing large datasets and has the potential to streamline future studies of cloud evolution and atmospheric processes on Venus. By applying our method to datasets spanning multiple time periods, researchers could systematically track the formation, propagation and dissipation of clouds features over time, which would have been difficult to achieve with manual classification methods due to the large volume of data. In the context of the broader field of planetary science, our technique could of course also be applied to other planetary bodies with similar atmospheric phenomena such as Titan.

Looking ahead, the rapid advancements in machine learning and artificial intelligence in recent times allow us to remain optimistic that the data limitations and methodological challenges may be overcome in the future. New missions that extend the amount of available data or innovative model architectures should also unlock more effective classification strategies for Venus' clouds. This endeavor not only enhances our understanding of atmospheric phenomena on Venus but also contributes to the broader field of planetary science by refining our methodological toolkit for extraterrestrial atmospheric analysis.

CRediT authorship contribution statement

J. Mittendorf: Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Data curation. **K. Molaverdikhani:** Writing – review & editing, Supervision, Conceptualization. **B. Ercolano:** Writing – review & editing, Supervision. **A. Giovagnoli:** Writing – review & editing, Methodology, Investigation, Conceptualization. **T. Grassi:** Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The code, a link to the encoder model checkpoints and instructions for preparing the data are available in a GitHub repository at: https://github.com/jmittendo/planet-patch-classifier.

Acknowledgments

The authors acknowledge the Principal Investigator W. Markiewicz (MPAe, Lindau) of the *VMC* instrument onboard the *Venus Express* mission for providing datasets in the archive. Datasets of the *VMC* instrument have been downloaded from the *ESA Planetary Science Archive* (http://archives.esac.esa.int/psa).

PDS data products from the *Venus Climate Orbiter Akatsuki UVI* calibrated dataset (Murakami et al., 2017) were used for the results in this paper.

Thanks are extended to the *ORIGINS Data Science Lab* (*ODSL*) of the *ORIGINS Excellence Cluster*, funded by the *Deutsche Forschungsgemeinschaft* (*DFG*, German Research Foundation), Germany under Germany's Excellence Strategy – EXC 2094 – 390783311, for providing hardware support during the development of this project.

References

Chen, T., Kornblith, S., Norouzi, M., Hinton, G.E., 2020. A simple framework for contrastive learning of visual representations. In: Proceedings of the 37th International Conference on Machine Learning. In: Proceedings of Machine Learning Research, vol. 119, PMLR, pp. 1597–1607, URL: http://proceedings.mlr.press/v119/chen20j.html.

Dev, S., Lee, Y.H., Winkler, S., 2015. Categorization of cloud image patches using an improved texton-based approach. In: 2015 IEEE International Conference on Image Processing. ICIP, pp. 422–426. doi:10.1109/ICIP.2015.7350833.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition. CVPR, pp. 770–778. doi:10.1109/CVPR.2016.90.

Hoang, V.T., 2020. Cloud-ImVN 1.0 [dataset]. Mendeley Data, V2. doi:10.17632/vwdd9grvdp.2.

Kurihana, T., Foster, I., Willett, R., Jenkins, S., Koenig, K., Werman, R., Lourenco, R.B., Neo, C., Moyer, E., 2019. Cloud classification with unsupervised deep learning. In: Proceedings of the 9th International Workshop on Climate Informatics: CI 2019. pp. 229–233. doi:10.5065/y82j-f154.

Limaye, S.S., 1984. Morphology and movements of polarizations features on Venus as seen in the pioneer Orbiter Cloud Photopolarimeter data. Icarus 57 (3), 362–385. doi:10.1016/0019-1035(84)90124-6, URL: https://www.sciencedirect.com/science/article/pii/0019103584901246.

Limaye, S.S., Watanabe, S., Yamazaki, A., Yamada, M., Satoh, T., Sato, T.M., Nakamura, M., Taguchi, M., Fukuhara, T., Imamura, T., Kouyama, T., Lee, Y.J., Horinouchi, T., Peralta, J., Iwagami, N., Hashimoto, G.L., Takagi, S., Ohtsuki, S., ya Murakami, S., Yamamoto, Y., Ogohara, K., Ando, H., ichiro Sugiyama, K., Ishii, N., Abe, T., Hirose, C., Suzuki, M., Hirata, N., Young, E.F., Ocampo, A.C., 2018. Venus looks different from day to night across wavelengths: morphology from Akatsuki multispectral images. Earth Planets Space 70 (1), 24. doi:10.1186/s40623-018-0789-5.

Markiewicz, W., Titov, D., Ignatiev, N., Keller, H., Crisp, D., Limaye, S., Jaumann, R., Moissl, R., Thomas, N., Esposito, L., Watanabe, S., Fiethe, B., Behnke, T., Szemerey, I., Michalik, H., Perplies, H., Wedemeier, M., Sebastian, I., Boogaerts, W., Hviid, S., Dierker, C., Osterloh, B., Böker, W., Koch, M., Michaelis, H., Belyaev, D., Dannenberg, A., Tschimmel, M., Russo, P., Roatsch, T., Matz, K., 2007. Venus Monitoring Camera for Venus Express. Planet. Space Sci. 55 (12), 1701–1711. doi:10.1016/j.pss.2007.01.004, URL: https://www.sciencedirect.com/science/article/pii/S0032063307000086. The Planet Venus and the Venus Express Mission, Part 2.

Minnaert, M., 1941. The reciprocity principle in lunar photometry. Astrophys. J. 93, 403–410. doi:10.1086/144279.

Molaverdikhani, K., McGouldrick, K., Esposito, L.W., 2012. The abundance and vertical distribution of the unknown ultraviolet absorber in the venusian atmosphere from analysis of Venus Monitoring Camera images. Icarus 217 (2), 648–660. doi:10.1016/j.icarus.2011.08.008, URL: https://www.sciencedirect.com/science/article/pii/S0019103511003174. Advances in Venus Science.

Murakami, S., Ogohara, K., Takagi, M., Kashimura, H., Yamada, M., Kouyama, T., Horinouchi, T., Imamura, T., 2018. Venus Climate Orbiter Akatsuki UVI longitude-latitude map data v1.1 [dataset]. JAXA data archives and transmission system.

Murakami, S., Yamada, M., Yamazaki, A., McGouldrick, K., Yamamoto, Y., Hashimoto, G.L., 2017. Venus Climate Orbiter Akatsuki UVI calibrated data, VCO-V-UVI-3-CDR-V1.0 [dataset]. NASA Planetary Data System. doi:10.17597/isas.darts/vco-00003.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S., 2019. PyTorch: An imperative style, high-performance deep learning

library. In: Advances in Neural Information Processing Systems. Vol. 32, Curran Associates, Inc., URL: https://proceedings.neurips.cc/paper_files/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. 12, 2825–2830.

Peralta, J., Iwagami, N., Sánchez-Lavega, A., Lee, Y.J., Hueso, R., Narita, M., Imamura, T., Miles, P., Wesley, A., Kardasis, E., Takagi, S., 2019. Morphology and dynamics of Venus's middle clouds with Akatsuki/IR1. Geophys. Res. Lett. 46 (5), 2399–2407. doi:10.1029/2018GL081670, URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2018GL081670.

Piccialli, A., Titov, D., Sanchez-Lavega, A., Peralta, J., Shalygina, O., Markiewicz, W., Svedhem, H., 2014. High latitude gravity waves at the Venus cloud tops as observed by the Venus Monitoring Camera on board Venus Express. Icarus 227, 94–111. doi:10.1016/j.icarus.2013.09.012, URL: https://www.sciencedirect.com/science/article/pii/S0019103513003928.

Roatsch, T., Markiewicz, W., 2015. VEX-VMC to planetary science archive interface control document (EAICD). URL: https://archives.esac.esa.int/psa/ftp/VENUS-EXPRESS/VMC/VEX-V-VMC-3-RDR-V3.0/DOCUMENT/VMC_EAICD.PDF.

Rossow, W.B., Del Genio, A.D., Limaye, S.S., Travis, L.D., Stone, P.H., 1980. Cloud morphology and motions from Pioneer Venus images. J. Geophys. Res. Space Phys. 85 (A13), 8107–8128. doi:10.1029/JA085iA13p08107, URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/JA085iA13p08107.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L., 2015. ImageNet large scale visual recognition challenge. Int. J. Comput. Vis. 115 (3), 211–252. doi:10.1007/s11263-015-0816-y.

Titov, D.V., Markiewicz, W.J., Ignatiev, N.I., Song, L., Limaye, S.S., Sanchez-Lavega, A., Hesemann, J., Almeida, M., Roatsch, T., Matz, K.-D., Scholten, F., Crisp, D., Esposito, L.W., Hviid, S.F., Jaumann, R., Keller, H.U., Moissl, R., 2012. Morphology of the cloud tops as observed by the Venus Express Monitoring Camera. Icarus 217 (2), 682–701. doi:10.1016/j.icarus.2011.06.020, URL: https://www.sciencedirect.com/science/article/pii/S0019103511002375. Advances in Venus Science.

van der Maaten, L., Hinton, G., 2008. Visualizing data using t-SNE. J. Mach. Learn. Res. 9 (86), 2579–2605, URL: http://jmlr.org/papers/v9/vandermaaten08a.html.

Van Gansbeke, W., Vandenhende, S., Georgoulis, S., Proesmans, M., Van Gool, L., 2020. SCAN: Learning to classify images without labels. In: Computer Vision – ECCV 2020. Springer International Publishing, Cham, pp. 268–285.

Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S.J., Brett, M., Wilson, J., Millman, K.J., Mayorov, N., Nelson, A.R.J., Jones, E., Kern, R., Larson, E., Carey, C.J., Polat, İ., Feng, Y., Moore, E.W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E.A., Harris, C.R., Archibald, A.M., Ribeiro, A.H., Pedregosa, F., van Mulbregt, P., SciPy 1.0 Contributors, 2020. SciPy 1.0: Fundamental algorithms for scientific computing in Python. Nat. Methods 17, 261–272. doi:10.1038/s41592-019-0686-2.

Ward, Jr., J.H., 1963. Hierarchical grouping to optimize an objective function. J. Amer. Statist. Assoc. 58 (301), 236–244. doi:10.1080/01621459.1963.10500845.

Yamazaki, A., Yamada, M., Lee, Y.J., Watanabe, S., Horinouchi, T., Murakami, S.-y., Kouyama, T., Ogohara, K., Imamura, T., Sato, T.M., Yamamoto, Y., Fukuhara, T., Ando, H., Sugiyama, K.-i., Takagi, S., Kashimura, H., Ohtsuki, S., Hirata, N., Hashimoto, G.L., Suzuki, M., Hirose, C., Ueno, M., Satoh, T., Abe, T., Ishii, N., Nakamura, M., 2018. Ultraviolet imager on Venus orbiter Akatsuki and its initial results. Earth Planets Space 70 (1), 23. doi:10.1186/s40623-017-0772-6.

You, Y., Gitman, I., Ginsburg, B., 2017. Large batch training of convolutional networks. doi:10.48550/arXiv.1708.03888, arXiv:1708.03888.