remote sensing

Article

Collection of a Hyperspectral Atmospheric Cloud Dataset and Enhancing Pixel Classification through Patch-Origin Embedding

Hua Yan [1], Rachel Zheng [1], Shivaji Mallela [1], Randy Russell [2] and Olcay Kursun [1,*]

1 Department of Computer Science and Computer Information Systems, Auburn University at Montgomery, Montgomery, AL 36117, USA; hyan@aum.edu (H.Y.); rzheng@aum.edu (R.Z.); smallel3@aum.edu (S.M.)
2 Department of Chemistry, Auburn University at Montgomery, Montgomery, AL 36117, USA; rrussell@aum.edu
* Correspondence: okursun@aum.edu

Abstract: Hyperspectral cameras collect detailed spectral information at each image pixel, contributing to the identification of image features. The rich spectral content of hyperspectral imagery has led to its application in diverse fields of study. This study focused on cloud classification using a dataset of hyperspectral sky images captured by a Resonon PIKA XC2 camera. The camera records images using 462 spectral bands, ranging from 400 to 1000 nm, with a spectral resolution of 1.9 nm. Our preliminary/unlabeled dataset comprised 33 parent hyperspectral images (HSI), each a substantial unlabeled image measuring 4402-by-1600 pixels. With the meteorological expertise within our team, we manually labeled pixels by extracting 10 to 20 sample patches from each parent image, each patch consisting of a 50-by-50 pixel field. This process yielded a collection of 444 patches, each categorically labeled into one of seven cloud and sky condition categories. To embed the inherent data structure while classifying individual pixels, we introduced an innovative technique to boost classification accuracy by incorporating patch-specific information into each pixel's feature vector. The posterior probabilities generated by these classifiers, which capture the unique attributes of each patch, were subsequently concatenated with the pixel's original spectral data to form an augmented feature vector. We then applied a final classifier to map the augmented vectors to the seven cloud/sky categories. The results compared favorably to the baseline model devoid of patch-origin embedding, showing that incorporating the spatial context along with the spectral information inherent in hyperspectral images enhances the classification accuracy in hyperspectral cloud classification. The dataset is available on IEEE DataPort.

Keywords: multi-spectral and hyperspectral remote sensing; cloud classification; feature vector augmentation; spatial contextual guidance; patch-origin embedding; transfer learning

check for updates

Citation: Yan, H.; Zheng, R.; Mallela, S.; Russell, R.; Kursun, O. Collection of a Hyperspectral Atmospheric Cloud Dataset and Enhancing Pixel Classification through Patch-Origin Embedding. *Remote Sens.* **2024**, *16*, 3315. https://doi.org/10.3390/rs16173315

Academic Editors: Levente Klein, Conrad M. Albrecht, Martin Werner and Salman Khan

Received: 4 July 2024
Revised: 29 August 2024
Accepted: 4 September 2024
Published: 6 September 2024

1. Introduction

Hyperspectral imagery (HSI) has applications in diverse areas, including agriculture, forestry, detection of mineral resources, land use classification for land management, and meteorology [1–5]. Airborne and spaceborne HSI have increasingly become powerful tools in atmospheric sciences for air quality analysis, measuring concentrations of water vapor and other atmospheric gases, and characterizing weather systems and clouds [4]. Ground-based hyperspectral sky imaging has also been used in many applications, including the retrieval of cloud microphysical and optical properties in both thin water clouds and cirrus clouds [6,7]. The use of ground-based hyperspectral sky imaging for the classification of clouds is an emerging field. Clouds of various types form at different levels in the Earth's atmosphere and have varying degrees of opacity to solar and terrestrial radiation. As a result, cloud type is an important factor in determining the flux of solar radiation at both the Earth's surface and the top of the atmosphere. The influence of cloud type on the Earth's radiation budget has been found to be as important as that of cloud amount.

Therefore, the regional and global climates are closely tied to the frequency of occurrence of different cloud types [8]. Major cloud genera are based on cloud height, morphology, and microphysical properties. Some of the major cloud genera, such as nimbostratus, infrequently occur at the location where our hyperspectral camera captures cloud/sky images to create the dataset for the study. The dataset we created emphasizes the small-scale (50 $\times$ 50 pixels) cloud/sky features that can be directly observed by a ground-based camera and related to the microphysical and radiative properties of clouds and a clear sky. It may be possible to deduce other cloud features related to cloud morphology from the analysis of the larger-scale images from which we took the small-scale samples, making it possible to identify cloud genera.

This study focused on the classification of regions within hyperspectral images into seven distinct cloud/sky categories. While each pixel in a hyperspectral image contains a wealth of spectral information, including textural details can improve the determination of membership in a particular cloud/sky category. Texture and the spatial extent of cloud elements are important factors in traditional methods for cloud classification. One approach to include texture would be to input a group of neighboring pixels into a classifier, such as a neural network. However, that approach does not necessarily optimize single-pixel classifier performance or feature extraction. A better method for feature extraction involves utilizing the contextual information contained in neighboring pixels within the images in a manner similar to positional embedding techniques recently popularized by transformers [9].

We propose a novel approach to pixel classification that includes the original spectral features but also enriches the data by incorporating contextual/textural information derived through the proposed augmentation algorithm. This algorithm, which we term "patch-origin embedding", augments the pixel feature vector with the posterior probabilities from several classifiers that identify the patch-origin of a pixel. Since each patch may represent a distinct texture, features that distinguish these textures or highlight their similarities can be beneficial. This enriched dataset is designed to enhance the distinctiveness of the spectral features, thereby improving the classifier's ability to discriminate between different cloud/sky types.

By comparing the classification performance on the original dataset against the augmented (enriched) dataset, this study aimed to demonstrate the potential benefits of our proposed method in enhancing the accuracy and reliability of hyperspectral cloud classification. The dataset is available on IEEE DataPort [10].

2. Collection of the Hyperspectral Dataset

This section describes the collection and post-processing of hyperspectral images and the sampling of pixels from those images to produce the dataset used in this study.

2.1. Image Data Acquisition

The Resonon PIKA XC2 hyperspectral camera used in this study operates as a scanning spectrometer. The camera first captures a small ribbon-like image of the sky within its integration field of view, and then it rotates clockwise around a vertical axis to capture an image of the abutting ribbon and appends it to the preceding image. It continuously rotates, captures, and appends until an entire image is accumulated. The result is an image of 4402 $\times$ 1600 pixels, with each pixel containing 462 bands of spectral data. The working principle of the Resonon camera is illustrated in Figure 1.

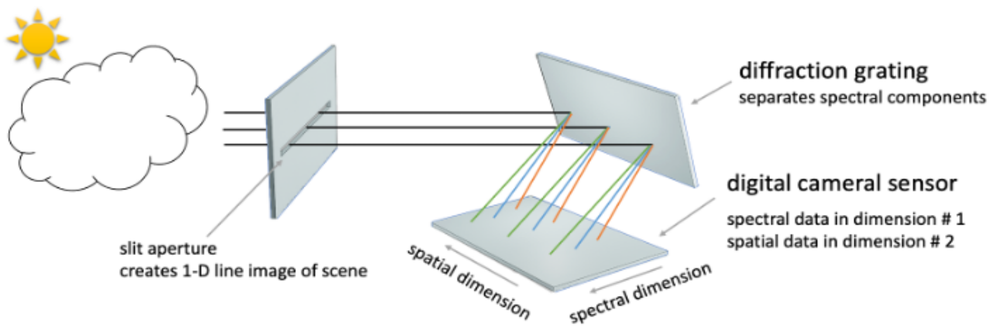
diffraction grating
separates spectral components
digital cameral sensor
spectral data in dimension # 1
spatial data in dimension # 2
slit aperture
creates 1-D line image of scene
spatial dimension
spectral dimension

Figure 1. Working principle of Resonon Hyperspectral Imager (inspired from [11]).

The camera is mounted on an Oben VH-R2 tilt head attached to a rotational scanning stage [12]. The tilt head maintains the vertical orientation of the camera at a stable and specific elevation angle during the scanning process. Figure 2 shows how the camera is set up. To seamlessly capture a high-quality image without blurring, the scan rate (SR), the number of degrees the stage pans per second, the frame rate (FR), and the number of individual exposures the camera captures per second must be synchronized. For our camera, the scan rate and frame rate were related by the following:

$$SR = 0.02035\,FR \tag{1}$$

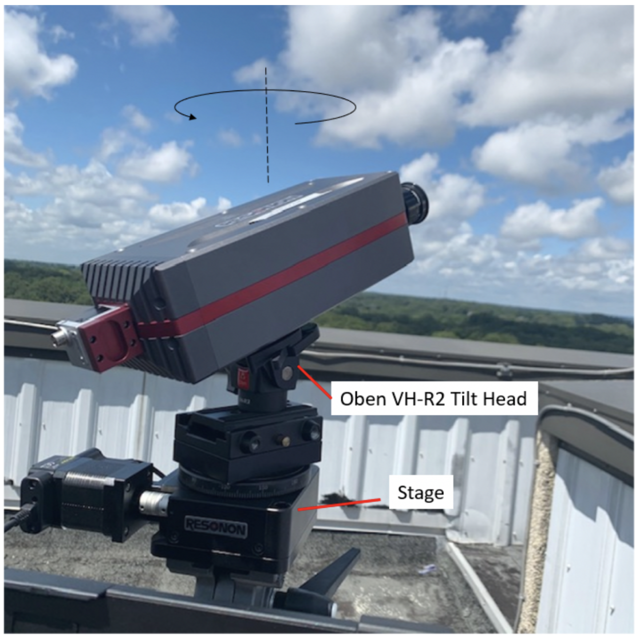

Figure 2. Resonon Pika XC2 camera mounted on a tilt head and attached to a rotational stage that captures sky images covering a 90-degree range in azimuth [12].

The camera captures spectral data across 462 channels, spanning wavelengths from 400 nm to 1000 nm, with measurements taken at intervals of approximately 1.31 nm and a spectral resolution of 1.9 nm FWHM (full width at half maximum) [13]. This configuration ensures a detailed spectral resolution across the visible to near-infrared spectrum. Thus, our HSI dataset has each pixel containing 462 channels, numbered from 0 to 461. The channels are organized in ascending order by wavelength, which supports a more intuitive analysis and data processing. The camera is equipped with an objective lens with a 17 mm focal length, giving a field of view of 30.8 degrees distributed over 1600 spatial pixels. The large number of spatial pixels and the small integration field of view of 0.71 mrad give the camera a spatial resolution high enough to reveal fine details in the cloud structure.

2.2. Image Data Processing Using Radiance Calibration and Normalization

Raw images (e.g., Figure 3) captured by the PIKA XC2 were calibrated in a post-processing step to obtain images for which the digital number of each wavelength channel is equal to monochromatic radiance in units of *μflicks*. The calibration was performed using Resonon's Spectronon software (version 3.5.5) and calibration data from Resonon.

After the calibration, to address variations in total radiance, which can arise from factors like atmospheric conditions and sensor sensitivity, the spectra were normalized using a method that involves dividing each value of monochromatic radiance by the radiance at a specific wavelength (586 nm), which is proportional to the spectrally averaged radiance under all sky conditions. This approach utilizes the radiance at 586 nm as a proxy for the spectrally averaged radiance, as it simplifies the process, and is effective for our purpose. The normalization formula is given by the following:

$$R'(\lambda) = \frac{R(\lambda)}{R(586\,\text{nm})} \tag{2}$$

where $R'(\lambda)$ represents the normalized radiance at wavelength λ, and $R(\lambda)$ is the monochromatic radiance at wavelength λ.

Normalization emphasizes the chromatic content of the data. The sky radiance in a clear sky regions is dominated by short-wavelength light originating from Rayleigh scattering by air molecules. In contrast, the sky radiance in cloudy regions is dominated by Mie scattering by cloud particles, which occurs more uniformly with wavelength [14–18]. By emphasizing the chromatic content of the images, the chromatic differences between dense clouds, thin or semi-transparent clouds, and clear sky are also highlighted. Figure 4 shows exemplary patches for each cloud/sky category. Figures 5–10 show the contribution of the normalization to the comparability of spectral signatures by standardizing variations in total radiance. Although normalization using the L2 (or L1) norm could be considered for multispectral imaging due to its ability to account for variations in total radiance across multiple wavelengths, we decided against using it. This decision was based on its computational complexity and the greater benefits derived from focusing on specific critical wavelengths tailored to potential multispectral applications. The L2 norm method would be represented as follows:

$$R'(\lambda) = \frac{R(\lambda)}{\sqrt{\sum_\lambda R(\lambda)^2}} \tag{3}$$

However, instead of the costly integration over the full set of wavelengths in the denominator, by identifying 586 nm as a key wavelength and using it in Equation (2), we reduce computational requirements and enhance our ability to generalize our approach to more cost-effective multispectral studies in the future.

2.3. Dataset Preparation

A total of 33 images, referred to as parent images, were selected for preparing the dataset for this research project. Using Resonon's Spectronon software, we extracted 10 to 20 sample patches from each parent image, each with a 50 × 50 pixel field. Based on our experiments and observations, we decided to use a sample patch size of 50 × 50 pixels to exclude many unwanted pixels surrounding the cloud of study in the parent image. As a result, we obtained cleaner pixel data for each cloud/sky category used for this study. Figure 3 presents a parent image with 10 selected sample patches, highlighted by red squares, indicating their respective locations.

Each sample patch was also saved as an image, referred to as a patch image. This process yielded a collection of 444 patches. These patches were manually labeled into seven cloud/sky categories based on the meteorological expertise within our team. Table 1 lists the seven cloud/sky categories used in this work and the number of patches in each category. The panels (a)–(g) of Figure 4 display exemplary patches from all seven sky/cloud categories in context.

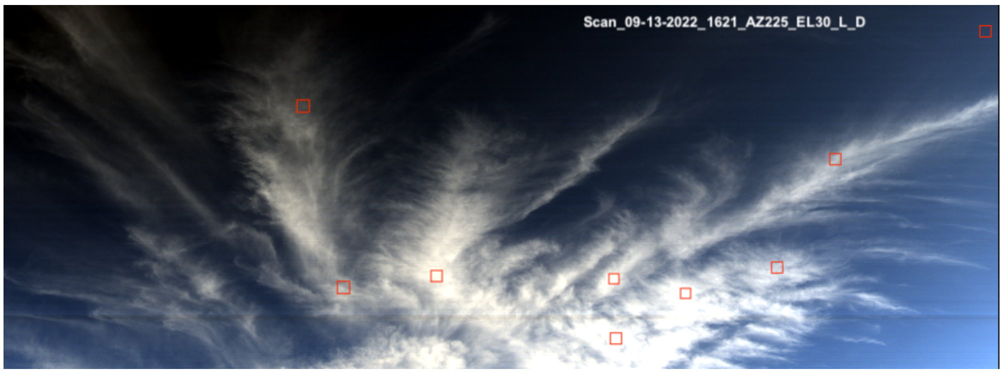

Figure 3. A sample parent image with some sample patches marked in red squares.

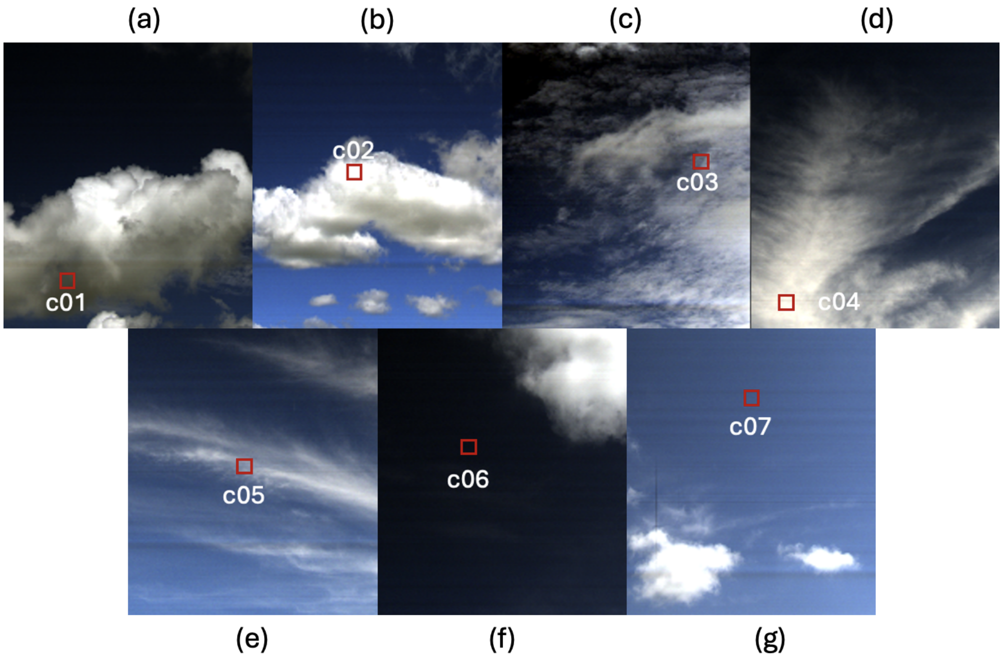

Figure 4. Sample patch examples for each cloud/sky category. (**a**) Dense dark cumuliform clouds (c01). (**b**) Dense bright cumuliform clouds (c02). (**c**) Semi-transparent cumuliform clouds (c03). (**d**) Dense cirroform clouds (c04). (**e**) Semi-transparent cirroform clouds (c05). (**f**) Low aerosol clear sky (c06). (**g**) Moderate/high aerosol clear sky (c07).

Although using the texture in the entire patch is the simplest approach, the advantage of HSI cameras lies in the richness of individual pixels, which is why we focused on pixel classification in this study. Once individual pixels can be classified most effectively, a patch-based classification study can follow, potentially achieving even higher accuracies. Our study leveraged spatial information through "patch-origin embedding", which enhances pixel representation by capitalizing on each pixel's patch membership. This approach, detailed in Section 3, utilizes predictor features indicative of the patch ID to enhance pixel representation without explicitly analyzing the entire 50×50 pixel field for texture. By concentrating on hyperspectral pixel classification, we leverage the detailed spectral information provided by HSI data. This method, which falls under the umbrella of transfer learning, is enabled by creating auxiliary classes using contextual guidance (spatial relations among pixels) and inspired by our past studies [19,20]. It can be extended to other applications within the field of remote sensing.

It is expected that each category of the labeled dataset has its signature. Figures 5, 7 and 9 display the spectral signatures of the three main categories: cumuliform (c01), cirroform (c04), and clear sky (c07). Each waveform represents the spectral data from a single pixel extracted from the central location (25,25) of a 50 × 50 patch for each category. Figures 6, 8 and 10 display the normalized spectral data for each category (as described in Equation (2)). In these six figures, the red dashed-line plots represent the average spectrum for each class, while the other colors illustrate three exemplary individual pixels. For instance, Figure 6 simplifies the signature so that it is more effectively processed by the downstream classifier, in contrast to the unnormalized spectra shown in Figure 5. Each plot in other colors represents the spectrum from a single pixel, with three such pixels provided as examples for each class.

Table 1. The cloud/sky categories and the distribution of our 444-patch dataset among these seven categories.

CategoryID	Category Description	Total Patches
c01	Dense Dark Cumuliform Cloud	54
c02	Dense Bright Cumuliform Cloud	79
c03	Semi-transparent Cumuliform Cloud	76
c04	Dense Cirroform Cloud	46
c05	Semi-transparent Cirroform Cloud	56
c06	Clear Sky—Low Aerosol Scattering (Dark)	68
c07	Clear Sky—Moderate to High Aerosol Scattering (Bright)	65
	Total	444

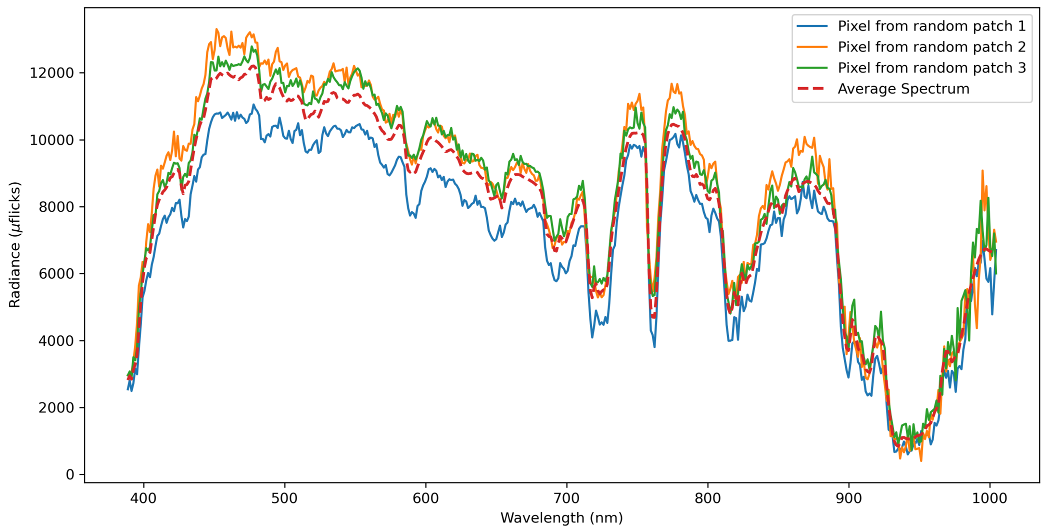

Figure 5. Spectra of three exemplary pixels obtained from three separate cumuliform cloud patches, each with 462 bands.

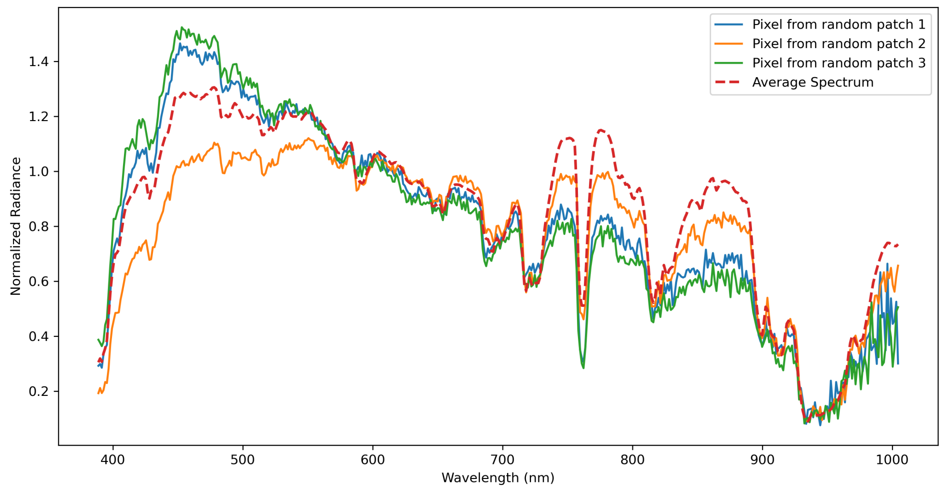

Figure 6. Normalized version of the cumuliform spectra of the three pixels and the class average of the normalized spectra.

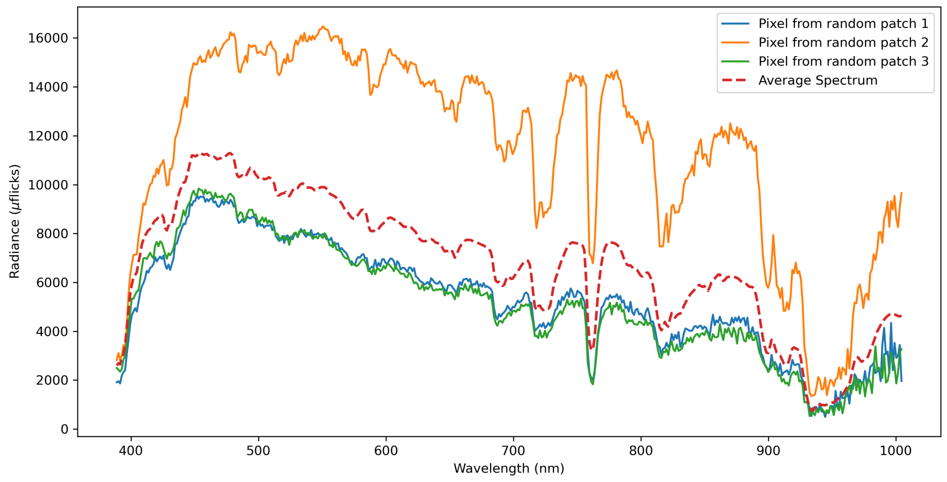

Figure 7. Spectra of three exemplary pixels obtained from three separate cirroform cloud patches, each with 462 bands.

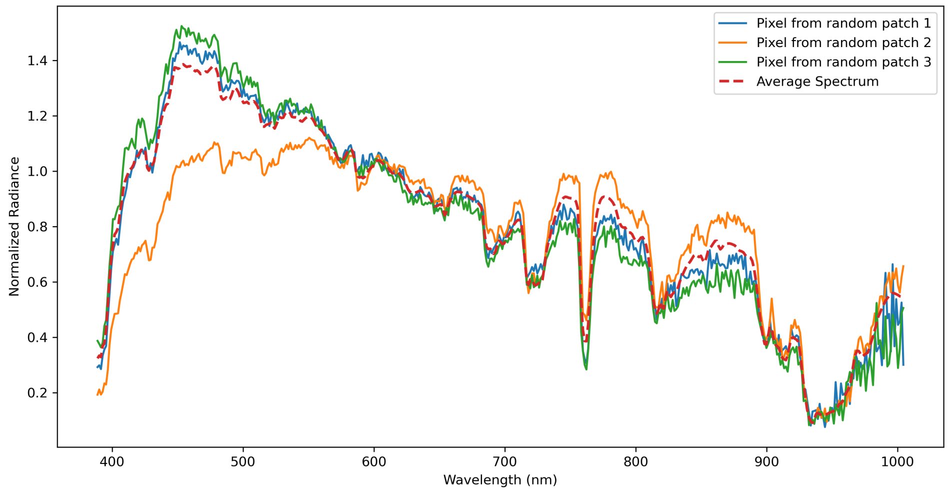

Figure 8. Normalized versions of the cirroform spectra of the three pixels and the class average of the normalized spectra.

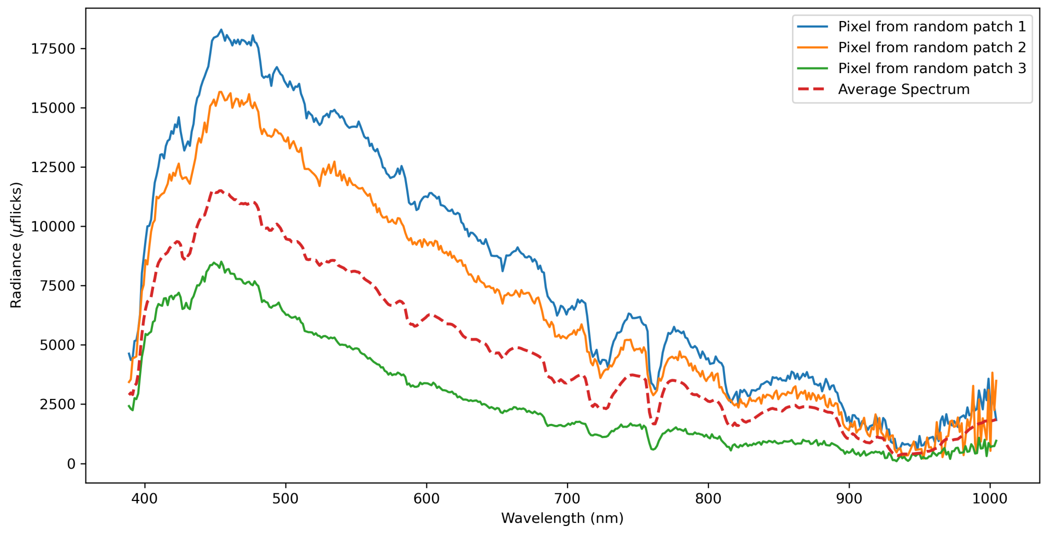

Figure 9. Spectra of three exemplary pixels obtained from three separate clear-sky patches, each with 462 bands.

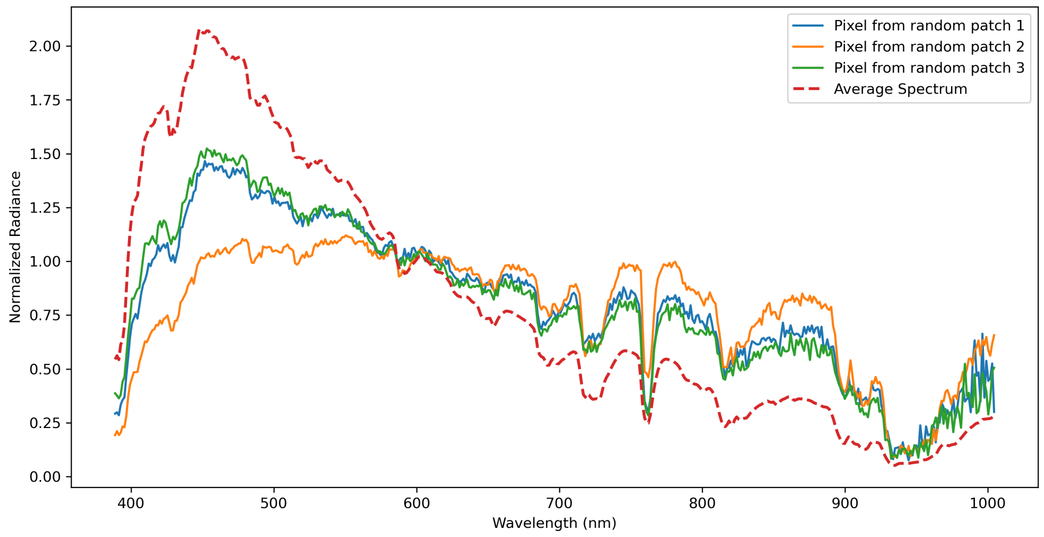

Figure 10. Normalized version of the clear-sky spectra of the three pixels and the class average of the normalized spectra.

We established a structured naming convention for our image files to ensure systematic organization and straightforward interpretation of our images. Although this dataset, including HSI image files encoded with extensive metadata, is available at [10], we provide a detailed explanation of our naming convention here. This clarification aims to make this paper a useful guide for users of the dataset and to facilitate its ease of use. The filenames for the parent image files are structured as follows:

```
SCAN_<date>_<time>_<azimuth>_<elevation>_<location>_<calibration>
```

where each component in the filename represents the following:

- `<date>`: The date the image was captured, formatted as MM-DD-YYYY.
- `<time>`: The time the image was captured, formatted as HHMM.
- `<azimuth>`: The azimuth angle of the camera at the beginning of the scan is in degrees, formatted as AZ###. AZ is the horizontal angle measured clockwise from North to the direction the camera was aimed at when the scan began.

- `<elevation>`: The elevation angle of the camera at the time of capture, formatted as EL##. EL is the vertical angle between the local horizon and the direction the camera was aiming at, measured in degrees. The elevation angle was constant during a given scan.
- `<location>`: The location of the camera during image capture, either Ground (G) or roof of the AUM library (L).
- `<calibration>`: The indicator of whether the image has undergone radiance calibration, denoted as calibrated to spectral ra<u>D</u>iance (D) or ra<u>W</u> (W).

Extensions for these files are `.bip` for the data and `.bip.hdr` for the header. The .bip extension indicates a binary file with interleaving by pixel. For each patch image file, we append the following suffix before the file extension to provide specific information about the patch:

`_<category>_<x_coord><y_coord><width><height>`

This suffix includes the following:

- `<category>`: The classification of the patch, formatted as c## (c01 through c07 corresponding to our seven categories).
- `<x_coord>, <y_coord>`: The x and y coordinates indicate the position of the patch within the parent image, each formatted as four digits.
- `<width>, <height>`: The dimensions of the patch in pixels, formatted as two digits each corresponding to the number of samples and the number of lines in the patch, respectively.

When displayed as a so-called waterfall image, the left-hand side of an image corresponds to those pixels with the highest elevation, i.e., the upward direction from the horizon is to the left of the displayed image. Pixels located at the top of the waterfall image have an azimuth equal to that at the beginning of the scan, i.e., the downward direction on the displayed image is counterclockwise from the beginning azimuth. The origin of the x, y coordinate system, used to identify pixel position, is located in the upper left-hand corner of the waterfall image. The x-coordinate increases to the right and is also referred to as the sample number since each pixel along a horizontal line was imaged during the same exposure of the integration field of view. The y-coordinate increases toward the bottom of the waterfall image and is also referred to as the line number since it numbers the exposures of the integration field of view sequentially. Figure 11 shows the coordinate system displayed with increasing elevation toward the top of the image and increasing azimuth to the right, i.e., the waterfall image is rotated ninety degrees clockwise. The suffix appended to the name of each patch image indicates the x and y coordinates of the upper left corner of the patch within the parent image when the image is viewed as a waterfall image.

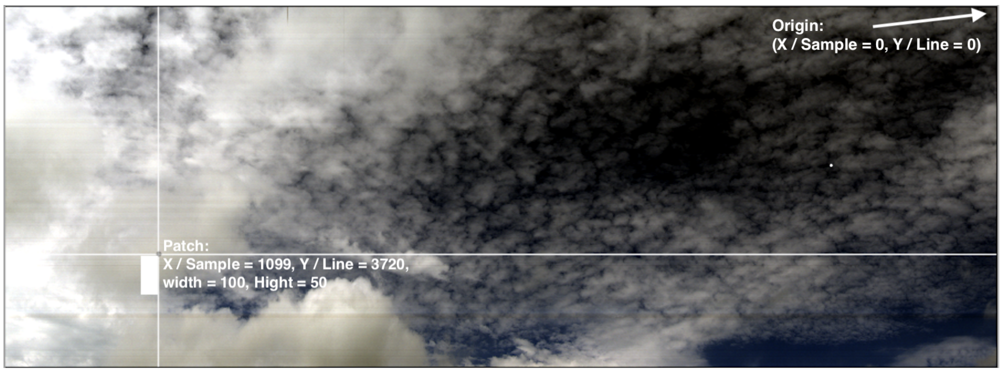
Origin:
(X / Sample = 0, Y / Line = 0)
Patch:
X / Sample = 1099, Y / Line = 3720,
width = 100, Hight = 50

Figure 11. Notations used for the origin of a parent image and the location and size of a patch in the parent image.

Figure 12 shows an example of a parent image file name demonstrating the naming convention. Figure 13 shows the extension added to the name of a parent image to produce the name of a patch selected from the parent image.

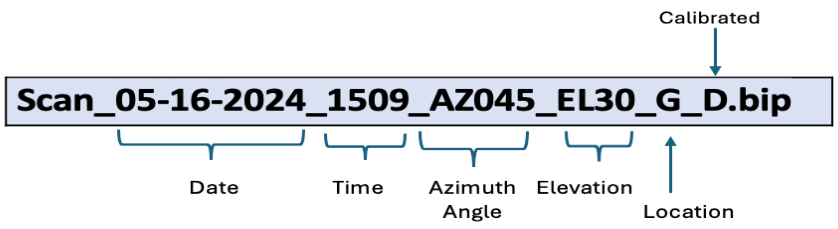

Figure 12. Parent image file naming convention of the dataset [10].

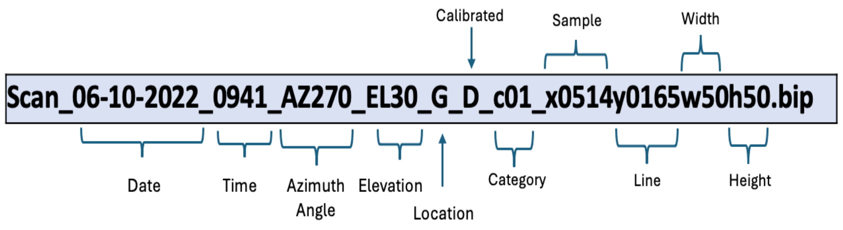

Figure 13. Patch image file naming convention of the dataset [10].

Although the whole fully-processed dataset is also available, to enhance data accessibility and facilitate straightforward spreadsheet-based predictive analysis, 100 pixels are randomly selected from each sample patch. This process yields a data matrix with dimensions of $44,400 \times 462$. In addition to the spectral channel data, each pixel (each row of the data matrix) also includes the category of sky/cloud to which it is classified, the ID of the parent image, and the ID of the patch image. These features enhance the granularity and contextual content of the dataset instances. All 444 patch images, with their corresponding appended information, are recorded in a CSV file and made available on IEEE DataPort [10]. The patch and parent images as HSI files (.bip and .hdr) are also available upon request. Each parent image contains a 6.36 GB .bip file and a 5 KB .hdr file. Each sample patch contains a 2.25 MB .bip file and a 5 KB .hdr file. As the parent files are very large image files, we also made 30 smaller unlabeled samples, each approximately 800×800 pixels in size, available on IEEE DataPort [21].

3. Proposed Classification Method with Patch-Origin Embedding

The proposed method aims to enhance classification accuracy by integrating patch-specific information into each pixel's feature vector. This method generates posterior probabilities for each pixel from classifiers trained on these patches, which are combined with the original spectral data to create an enriched feature vector. These vectors are subsequently merged and then classified into the seven categories using a final classifier. We tested logistic regression (LR) and convolutional neural networks (CNNs) to enrich the dataset. The enrichment was achieved by LR and CNN using the posterior probabilities as features and deriving features from the hidden layers of CNNs that are trained to predict those posteriors. There are alternative methods for generating such patch-embedding features that we experimented with in this study. These features make explicit the discriminative information in a pixel's spectrum. We called adding these features to the original dataset "patch-origin embedding". The rationale for relying on such features is rooted in transfer learning in deep learning and positional embedding in transformers. The models trained on the base task to predict the original patch from which each pixel is derived serve as the base task in the parlance of transfer learning [20,22,23]. Once they are learned, they enrich the original dataset, and together, they are used to train-test the target task [20,23,24], which in our application is the supervised 7-class cloud/sky classification task. In this section, we review LR and the computation for the posterior probabilities of patch-origins and describe how the feature set is enriched. We also test CNNs for the same task of modeling patch-origin prediction, as detailed in the Experimental Results. After the enrichment, a Random Forest (RF) classifier is used as one of the target task classifiers. RF is known to have high performance and simplicity [25–28]. As detailed in the Results section, LR and RF are applied to both the original and enriched datasets, and we compared their performances. The RF classifier has proven to be a robust ensemble of decision trees [28,29].

In its most basic form, the proposed classification method with patch-origin embedding employs LR for linear classification. LR, in its application to binary classification (i.e., $K = 2$), models the probability that a given input belongs to the class labeled as "1" (as opposed to class "0"). This is achieved by using a logistic function to map the predicted values to probabilities. The logistic function, also known as the sigmoid function, is defined as follows:

$$\sigma(z) = \frac{1}{1 + e^{-z}}, \tag{4}$$

where z is the linear combination of the input features ($\mathbf{x}$) weighted by the coefficients ($\boldsymbol{\beta_i}$) from the model, expressed as $z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$. In summary, LR calculates a boundary line between the two classes for binary classification. This boundary line's probability is 0.5, which exemplifies an equal likelihood of the data belonging to either class. If $p > 0.5$, it resides on the positive side of the boundary line for class 1, symbolizing a greater probability of being related to class 1. For $p < 0.5$, it is more likely to belong to class 0. Thus, the probability of the negative residing data value to class 0 would be the complement of the probability of residing in class 1 [27].

The logistic regression model, thus, predicts the probability p that an observation belongs to the class 1 as follows:

$$p = \sigma(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n). \tag{5}$$

This probability p can be interpreted as the likelihood that a given pixel belongs to a cloud/sky category based on its spectral characteristics. Logistic regression is particularly advantageous in scenarios where a clear probabilistic interpretation is required, making it suitable for classifying pixels in hyperspectral images where each pixel's class membership (such as cloud/sky category) needs to be probabilistically quantified.

For scenarios involving multiple classes (i.e., $K > 2$), the binary logistic regression model is extended to multinomial logistic regression, also known as softmax regression. This model handles multiple categories by modeling the probability that a given input belongs to each class. The decision rule for multinomial logistic regression involves computing a set of probabilities for each class using the softmax function, defined as

$$P(y = k|\mathbf{x}) = \frac{e^{\mathbf{x}^\top \beta_k}}{\sum_{j=1}^{K} e^{\mathbf{x}^\top \beta_j}}, \tag{6}$$

where $P(y = k|\mathbf{x})$ is the probability that the observation $\mathbf{x}$ belongs to class k, β_k is the coefficient vector for class k, $\mathbf{x}^\top \beta_k$ is the dot product of the transpose of the input pixel feature vector $\mathbf{x}$ and the coefficient vector β_k, and K is the total number of classes.

The predicted class label $\hat{y}$ for an observation is then given by the class that has the highest probability:

$$\hat{y} = \arg\max_k P(y = k|\mathbf{x}), \tag{7}$$

where $\arg\max_k$ denotes the argument of the maximum, which identifies the class k that maximizes the probability $P(y = k|\mathbf{x})$.

Rather than applying $\arg\max_k$ to the logistic regression output for obtaining a nominal value for decision-making, our study utilizes the calculated probabilities as posterior probabilities (each pixel's likelihood of belonging to each one of the K patches). We employ both a single-shot version, with K set to the number of all patches in the training set, and a combination of multiple runs with smaller K values and, thus, fusing posterior probabilities from multiple such K-class classifications. Specifically, we perform a procedure where K

classes are randomly selected, and we compute the merged posterior probabilities over these classes. This process is repeated N times to ensure generalization in the predictions. For our study, we set $K = 30$ and $N = 20$, allowing us to capture a broad spectrum of scenarios and reduce the influence of any single random selection of classes. The specific details of incorporation of the contextual information can be varied; to that end, we also performed another alternative in which we used the shattering in VC-dimension: we randomly picked a set of patches as class-1 and the rest as class-0 and repeated such random splits N times to ensure there is rich enough contextual information embedded into the feature vector.

The enhancement process involves stacking the posterior probabilities $p_{i,k}$ for each pixel i across multiple classes k and potentially across multiple model runs (iterations). For the multi-selection approach where N random selections of K classes are considered, the stacked vector for each pixel i is constructed as follows:

$$\mathbf{v}_i = \left[\mathbf{x}_i, \mathbf{p}_i^{(1)}, \mathbf{p}_i^{(2)}, \ldots, \mathbf{p}_i^{(N)}\right], \tag{8}$$

where

- $\mathbf{x}_i$ is the original spectral feature vector of pixel i.
- $\mathbf{p}_i^{(n)}$ represents the output of a softmax (or hidden) layer of the network for pixel i obtained from the n-th run. This vector contains the probabilities for each class k, which are computed by the network. Specifically,

  $$\mathbf{p}_i^{(n)} = \left[p_{i,1}^{(n)}, p_{i,2}^{(n)}, \ldots, p_{i,K}^{(n)}\right],$$

  where $p_{i,k}^{(n)}$ is the probability of pixel i belonging to class k in the n-th run, and K is the total number of classes.
- $\mathbf{v}_i$ is the enhanced feature vector for pixel i, created by concatenating the original feature vector $\mathbf{x}_i$ with the hidden layer outputs from all N runs.

These enhanced vectors $\mathbf{v}_i$ are then utilized as the input for the final classifier, such as a Random Forest. The RF utilizes decision trees in its evaluation to produce a probability of belonging. At each level of the decision tree, the data are tested and branch onto the next corresponding node that matches the evaluation outcome. This process continues until it ends at a leaf node, where further testing will guarantee the same results. It is crucial to note that decision trees do not assume a linear relationship and constantly adapt the discriminate tests accordingly. The combined results of multiple decision trees, the most frequent result or majority vote, produce a probability of belonging [27,30]. This method ensures that each classifier not only utilizes the inherent spectral characteristics of each pixel but also leverages the learned probabilistic distinctions between patches, thereby capturing both the spectral and contextual information crucial for accurate cloud classification.

4. Results

This section presents a concise analysis of the experimental results obtained by systematically varying the feature sets used to train our classifiers, thereby assessing the efficacy of different feature enrichment techniques in hyperspectral image classification. For training/testing the classifier models and for computing the embeddings, the Group Shuffle Split (GSS) technique was used to prevent class-label leakage between the training and test sets. GSS effectively ensures that all pixels from the same patch are allocated exclusively to either the training set or the test set, rather than being dispersed across both. We applied GSS to our dataset of 444 patches, designating 20% as the test size. This approach resulted in 355 patches being used for training in each experimental run and 89 patches for testing. Results were averaged over ten runs to enhance the reliability and robustness of our findings.

We start this section by demonstrating the challenging nature of the problem. Using an 80-20 train–test split with a decision tree on the raw (unnormalized) data for the initial evaluation yields low classification results with an accuracy of 71.90% out of 8900 test examples. The results summarized in Tables 2 and 3 indicate the difficulty of the classification problem. The imbalance of class labels and similarity among different classes (such as cumuliform classes c01/c02/c03 and cirroform classes c04/c05) contribute to the challenge, as evidenced by the varying precision, recall, F-measure, and Matthews Correlation Coefficient (MCC) values for each class. The results presented in Tables 2 and 3 benefit from additional examples from class c04, which helps improve its classification accuracy. However, distinguishing between the closely related cirroform classes c04 and c05 remains challenging, indicating the need for further refinement in future dataset updates [10].

Table 2. Detailed accuracy by class for the initial analysis using a simple decision tree classifier, without radiance normalization, and without contextually guided feature augmentation.

Class	TP Rate	FP Rate	Precision	F-Measure	MCC
c01	0.672	0.047	0.511	0.580	0.551
c02	0.802	0.036	0.848	0.824	0.782
c03	0.684	0.072	0.674	0.679	0.608
c04	0.672	0.029	0.723	0.697	0.664
c05	0.364	0.054	0.366	0.365	0.311
c06	0.847	0.061	0.737	0.788	0.744
c07	0.738	0.028	0.868	0.798	0.755
Weighted Avg.	0.683	0.047	0.675	0.676	0.631

Table 3. Confusion matrix on the sample test set for the initial analysis using a simple decision tree, without radiance normalization, and without contextually guided feature augmentation.

	Predicted Class						
Actual Class	**c01**	**c02**	**c03**	**c04**	**c05**	**c06**	**c07**
c01 = Cumuliform-1	403	28	96	57	16	0	0
c02 = Cumuliform-2	156	1443	191	2	7	0	1
c03 = Cumuliform-3	175	125	1094	50	152	2	2
c04 = Cirroform-1	20	1	20	605	254	0	0
c05 = Cirroform-2	35	4	193	123	255	87	3
c06 = Clear-Sky-1	0	0	21	0	12	1271	196
c07 = Clear-Sky-2	0	100	7	0	0	365	1328

In addition to the pixel classification study, we explored the effectiveness of RGB renders for classification by employing both custom-built and pre-trained deep learning models. We used these models to classify 50×50 patches from the dataset into seven classes, splitting the 444 image patches into training and test sets (80:20 ratio). Models trained from scratch achieved only about 40% accuracy. Testing several pre-trained models, including VGG16, EfficientNetV2, ResNetV250, InceptionV3, MobileNetV2, and DenseNet121, with dataset-specific fine-tuning improved accuracy but still fell short compared to our proposed pixel classification method. These experiments show that differentiating subclasses was particularly challenging. We reclassified the seven original classes into three broader categories: merging the first three cumuliform classes into one, combining the next two cirroform classes into another, and grouping the last two clear-sky classes into a final category. Using this revised three-class classification scheme, the custom-built CNN model (see Figure 14) achieved an average validation accuracy of 76% across ten data splits. Similarly, pre-trained models yielded validation accuracies between 74% and 76%, comparable to the custom-built model. The model that achieved the highest accuracy was structured as follows.

```
model = Sequential([
    Conv2D(32, 3, activation='relu', input_shape=(50,50,3)),
    MaxPooling2D((2,2)),
    Conv2D(64, 3, activation='relu'),
    MaxPooling2D(),
    Conv2D(128, 3, activation='relu'),
    MaxPooling2D(),
    Conv2D(256, 3, activation='relu'),
    MaxPooling2D(),
    Flatten(),
    Dense(256, activation='relu'),
    Dropout(0.3),
    Dense(128, activation='relu'),
    Dropout(0.3),
    Dense(7, activation='softmax') # 7 cloud/sky categories
])
```

Figure 14. CNN architecture used for patch classification using the RGB renders.

The proposed techniques, utilizing normalization and feature enrichment via contextual guidance, significantly improve these poor initial results as follows. Table 4 provides a detailed overview of these feature sets along with their respective dimensions. The original dataset comprises 462 dimensions. In addition to this, three other feature sets were utilized: CNN Hidden, CNN Posterior, and LR Posterior. These three are designed for incremental learning, which is advantageous for managing and processing large datasets effectively with $K = 30$ and $N = 20$ as outlined in Equation (8).

For our proposed incremental approach, for each one of the $N = 20$ classifiers, we randomly selected $K = 30$ classes from a total of 355 training patches at a time and trained a CNN to compute posterior probabilities for these classes. As Table 4 shows, the resulting 30×20 features are appended to the original 462-dimensional pixel feature vector, creating the "CNN Posterior" feature set with a total of $462 + 20 \times 30$ dimensions for each pixel. Alternatively, the "CNN Hidden" feature set utilizes features from the intermediate layers of the CNNs to enrich the original spectral data. This method incorporates spatial-contextual features extracted from hidden layers, enhancing the base 462 spectral features with additional data from 20 convolutional filters in a key hidden layer. Each filter produces 32 feature maps, resulting in a total of 1102 features. The CNN significantly augments the dataset with informative features crucial for precise classification by capturing and encoding complex spatial relationships within the hyperspectral images.

Both the CNN Hidden and CNN Posterior methods utilize features extracted by the CNN architecture illustrated in Figure 15. In this architecture, the final layer employs a softmax activation function to generate posterior probabilities of patch memberships for each input pixel, as utilized in Equation (8) for the CNN Posterior method. Additionally, we implemented the CNN Hidden approach, which extracts features from the hidden layers of the CNN. This approach is a well-established practice in transfer learning and deep learning, recognized for its effectiveness in feature extraction and serving as an excellent initialization point for downstream classification tasks [20,22,31]. Our CNN architecture in Figure 15 begins with an input layer designed to process hyperspectral images, which are input as 1D arrays (n_channels = 462) with dimensions corresponding to the number of spectral channels (it is designed to process a single pixel as input).

1. First Convolutional Layer: This layer uses 32 filters with a kernel size of 25 and a stride of 4. The use of a relatively large kernel size in the initial layer is intended to capture broader spectral features early in the network.
2. Pooling and Dropout: Following the first convolution, a max pooling layer reduces the output's spatial dimensions, which helps reduce computation and control overfitting. This is accompanied by a dropout layer with a rate of 0.2 to further regularize the model.
3. Subsequent Convolutional Layers: Additional convolutional layers with smaller kernels enhance the network's ability to learn more refined features. Each convolutional stage is followed by dropout and max pooling layers.
4. Global Max Pooling and Dense Layers: The final stages of the network flatten the output and transition to dense layers, culminating in a softmax output layer that classifies each pixel into one of the K classes.

```
# Build a CNN model
model = Sequential([
    Conv1D(32, kernel_size=25, strides=4, activation='relu',
       ↪ input_shape=(n_channels, 1)),  # Larger first conv layer
    Dropout(0.2),
    MaxPool1D(),
    Conv1D(32, kernel_size=3, strides=2, activation='relu'),
    Dropout(0.2),
    MaxPool1D(),
    Conv1D(32, kernel_size=3, strides=1, activation='relu'),
    GlobalMaxPooling1D(),
    Dense(64, activation='relu'),
    Dropout(0.2),  # Dropout
    Dense(K, activation='softmax')
])
```

Figure 15. CNN architecture used for feature extraction. Outputs from the network's GlobalMaxPooling layer serve as features to downstream classifiers, either LR or RF.

After the patch-origin embeddings are learned from the base task using the 355 training patches, we initiate the target (downstream) task, which involves classifying pixels into seven cloud/sky classes. For this, pixels from the 20% of patches reserved for testing are processed through the K-class classifiers to compute their resemblances to the training patches. This process aims to augment the test set in a manner analogous to the training set preparation, ensuring consistency in feature representation across both sets. These enhanced training and test sets are then input into a final classifier, either Logistic Regression (LR) or Random Forest (RF), chosen for their simplicity. This choice allows us to more effectively evaluate the enhancement's contributions without retraining a complex deep network for the seven-class classification task. The RF classifier operates on a majority voting principle among a number of decision trees, and we used the default configuration in the scikit-learn implementation that employs 100 trees [29,30].

Table 5 demonstrates that the classification accuracy suffers without the normalization process outlined in Equation (2). As shown in Table 6, the normalization addresses variations in total radiance and emphasizes chromatic content, leading to better differentiation between the signatures of cloud types and clear sky. The results presented in Table 6 evaluate and compare the performance of the classifiers with different feature sets based on the accuracy that measures the proportion of correctly classified instances and on Matthews Correlation Coefficient (MCC) that provides a more informative metric than accuracy, especially with imbalanced datasets. MCC balances for the bias created by classes with higher prior probabilities. An MCC of 1 indicates a perfect prediction, while an MCC of 0 indicates a prediction no better than random guessing based on the probability distribution of the priors [32].

Figure 16 shows four sample images from [21], which contains 30 large images (approximately 800 × 800 pixels) not used in the training set of this study. The first column displays the original image. For the middle and rightmost columns, we applied the trained model to classify each pixel individually. The middle column presents the classification results using the original features with a Random Forest classifier. The rightmost column illustrates the results using CNN Hidden features combined with a final Random Forest classifier. The middle column offers a more readily interpretable segmentation. However, the rightmost column reveals additional details concerning brightness that are missed in the middle column. In some cases (e.g., the third row), this column appears to blur the segmentation by adding details not discernible in the original render, although these details may be present in the richer data.

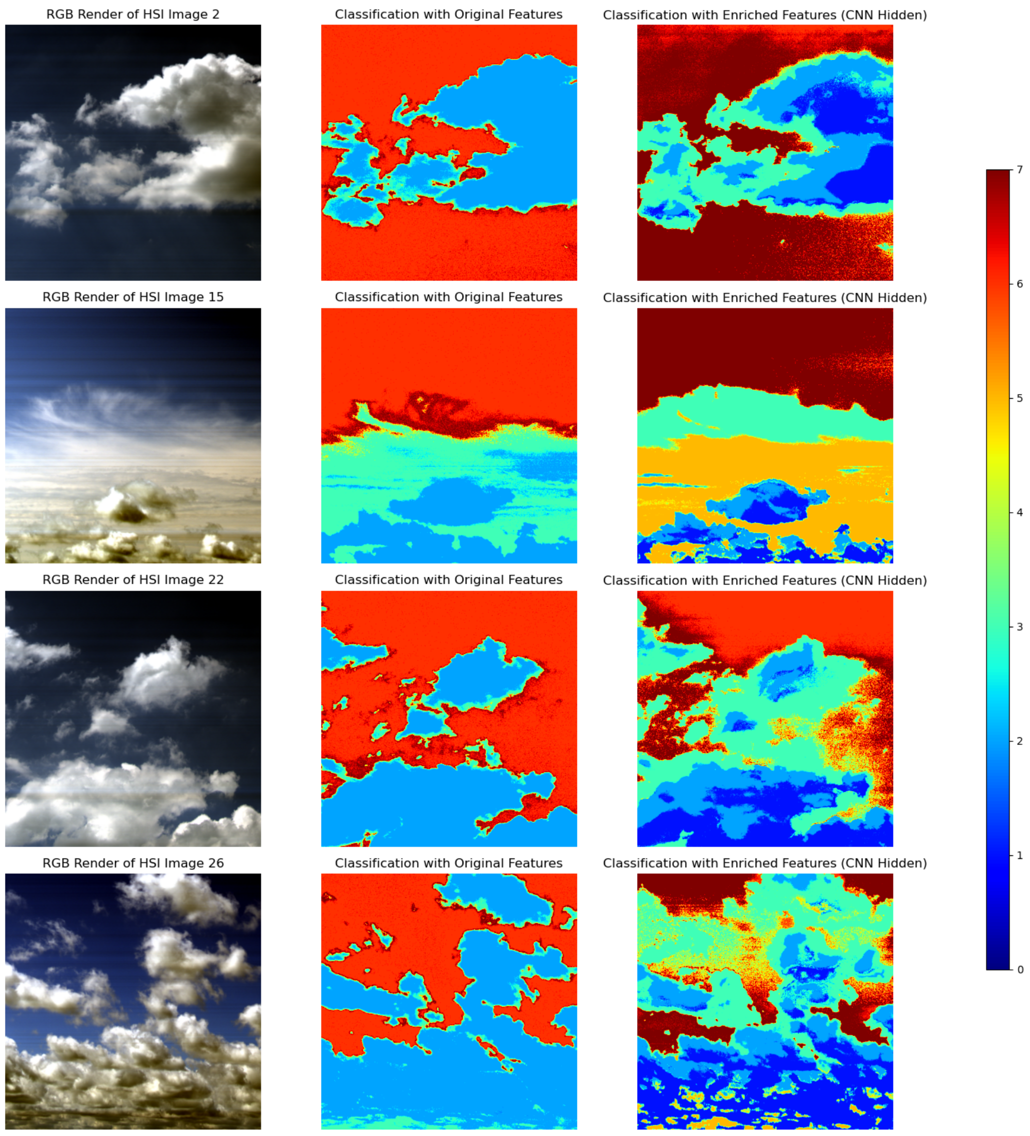
RGB Render of HSI Image 2
Classification with Original Features
Classification with Enriched Features (CNN Hidden)
RGB Render of HSI Image 15
Classification with Original Features
Classification with Enriched Features (CNN Hidden)
RGB Render of HSI Image 22
Classification with Original Features
Classification with Enriched Features (CNN Hidden)
RGB Render of HSI Image 26
Classification with Original Features
Classification with Enriched Features (CNN Hidden)
7
6
5
4
3
2
1
0

Figure 16. Classification results on sample parent images (from [21]), which are large images not included in the training dataset. The results demonstrate the performance of the classification model on new, unseen data.

Table 4. Overview of feature sets and their dimensionalities.

Feature Set	Description	Dimensions	Total Features
Original	Raw hyperspectral data	462	462
CNN Hidden	Original + CNN Hidden Layer Features	462 + 20 × 32	1102
CNN Posterior	Original + CNN Softmax Outputs	462 + 20 × 30	1062
LR Posterior	Original + Incremental LR Posteriors	462 + 20 × 30	1062

Table 5. Classifier performances without the radiance normalization.

Classifier	Feature Set	Accuracy (%)	MCC (%)
RF	Original	78.12 ± 2.88	74.51 ± 3.09
RF	LR Posterior	79.19 ± 2.50	75.75 ± 2.71
LR	Original	74.91 ± 3.81	70.73 ± 4.35
LR	LR Posterior	75.11 ± 3.36	70.95 ± 3.79

Table 6. Classifier performances with the radiance normalization on all feature sets.

Classifier	Feature Set	Accuracy (%)	MCC (%)
RF	Original	80.60	77.30
RF	CNN Hidden	82.90	79.96
RF	CNN Posterior	81.79	78.81
RF	LR Posterior	81.80	78.69
LR	Original	77.94	74.27
LR	CNN Hidden	80.95	77.65
LR	CNN Posterior	79.28	75.70
LR	LR Posterior	78.14	74.48

5. Discussion

Our study was supported by a rigorously collected, calibrated, and normalized dataset, enhanced with metadata such as azimuth, elevation, date–time, and patch-membership. Each image in our dataset underwent processing to ensure uniform radiance levels, with bounding boxes designating patches that were subsequently class-labeled for supervised learning in a seven-class classification problem. Now made publicly available, this dataset effectively facilitates the analysis of cloud/sky conditions, serving as a valuable resource for climate analysis and related HSI research.

The analysis of the dataset collected clearly demonstrates that both Random Forest (RF) and Logistic Regression (LR) classifiers benefit from the integration of enhanced feature sets, with improvements in accuracy, Matthews Correlation Coefficient (MCC), and visually more plausible classification results on the new large test images. The experimental results provide a comprehensive evaluation of two classifiers, RF and LR, where RF consistently outperforms LR in terms of both mean accuracy and MCC on the original feature set. This outcome is expected because of the nonlinear capabilities of RF [25]. Additionally, as the comparison between Tables 5 and 6 shows, the normalization process outlined in Equation (2) improves the classification accuracy of all classifiers on all features sets.

Overall, our results highlight that CNNs work favorably in extracting informative features, and their contextual enrichment significantly enhances pixel classification performance. Moreover, our contextual guidance approach, which was successful in this study, can be extended to other applications beyond remote sensing.

Our experimental results demonstrate that enriching the dataset with additional examples from undersampled classes improves classification accuracy. Therefore, as future work, we plan to enhance the dataset hosted in repository [10]. Future updates will include adding more classes and increasing the number of patches to provide a more comprehensive representation of cloud and sky conditions. These updates will be implemented within the same repository and made available as Open Access to ensure broader academic use in hyperspectral image analysis and cloud classification.

Author Contributions: H.Y. was responsible for project conceptualization, performed the data analysis and machine learning experiments, and oversaw research students. R.Z. focused on data acquisition, curation, and dataset analysis. S.M. developed and ran some of the deep learning programs. R.R. was involved in data acquisition, dataset design, labeling, and analysis. O.K. managed project administration and resource procurement, designed and supervised the machine learning experiments, and contributed to project conceptualization. All authors contributed to the review and editing of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Science Foundation, grant number 2003740.

Data Availability Statement: The hyperspectral images used in this study, along with their RGB renders and descriptive information, including day of the year, time of day, and the altitude and azimuth of the central pixel, are available on IEEE DataPort. The 7-class dataset is made available under the title CloudPatch-7 Hyperspectral Dataset at https://dx.doi.org/10.21227/fgb9-qs51 (accessed on 3 September 2024).

Acknowledgments: Thanks to our student researchers Derienne Black and Sameer Shaga, who helped with data acquisition and processing. Thanks to Kevser Kursun for helping with some of the figures.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

CNN	Convolutional Neural Network
GSS	Group Shuffle Split
HSI	Hyperspectral Image
LR	Logistic Regression Classifier
MCC	Matthews Correlation Coefficient
ML	Machine Learning
RF	Random Forest Classifier

References

1. Sellami, A.; Farah, M.; Riadh Farah, I.; Solaiman, B. Hyperspectral imagery classification based on semi-supervised 3-D deep neural network and adaptive band selection. *Expert Syst. Appl.* **2019**, *129*, 246–259. [CrossRef]
2. Lu, B.; Dao, P.; Lui, J.; He, Y. Recent Advances of Hyperspectral Imaging Technology and Applications in Agriculture. *Remote Sens.* **2020**, *12*, 2659. [CrossRef]
3. HySpex. Understanding Forests from A Hyperspectral Glance. Available online: https://www.hyspex.com/use-cases-application-notes/forestry-management/ (accessed on 31 May 2024).
4. Calin, M.A.; Calin, A.C.; Nicolae, D.N. Application of airborne and spaceborne hyperspectral imaging techniques for atmospheric research: Past, present, and future. *Appl. Spectrosc. Rev.* **2021**, *56*, 289–323. [CrossRef]
5. Mateen, M.; Wen, J.; Akbar, M.A. The role of hyperspectral imaging: A literature review. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *9*, 51–62. [CrossRef]
6. Hirsch, E.; Agassi, E.; Koren, I. Determination of optical and microphysical properties of thin warm clouds using ground based hyper-spectral analysis. *Atmos. Meas. Tech. Discuss.* **2011**, *4*, 7277–7335. [CrossRef]
7. Schäfer, M.; Bierwirth, E.; Ehrlich, A.; Heyner, F.; Wendisch, M. Application of ground-based hyperspectral imaging to retrieve ice crystal shape and fields of cirrus optical thickness. *Atmos. Meas. Tech. Discuss.* **2013**, *6*, 1201–1238.
8. Chen, T.; Rossow, W.B.; Zhang, Y. Radiative Effects of Cloud-Type Variations. *J. Clim.* **2000**, *13*, 264–286. [CrossRef]
9. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
10. Yan, H.; Zheng, R.; Boehm, B.; Shaga, S.; Black, D.; Parra, L.C.; Russell, R.; Kursun, O. CloudPatch-7 Hyperspectral Dataset. IEEE Dataport 2024. [CrossRef]
11. Resonon. Overview—Working Principle. Available online: https://resonon.gitlab.io/programming-docs/content/overview.html (accessed on 31 May 2024).

12. Nguyen, B. Impact of Feature Selection on Segmentation of Hyperspectral Images of Atmospheric Clouds. Master's Thesis, Auburn University at Montgomery, Montgomery, AL, USA, 2023.
13. Resonon. Resonon Product, Pika XC2. Available online: https://resonon.com/Pika-XC2 (accessed on 31 May 2024).
14. Government of Canada. Interactions with the Atmosphere. Canada.ca, Natural Resources Canada, Remote Sensing Tutorials, 2016. Data Modified: 2 March 2016. Available online: https://natural-resources.canada.ca/maps-tools-and-publications/satellite-imagery-elevation-data-and-air-photos/tutorial-fundamentals-remote-sensing/introduction/interactions-the-atmosphere/14635 (accessed on 3 July 2024).
15. University of Illinois at Urbana-Champaign. Scattering of Light. University of Illinois at Urbana-Champaign, Department of Atmospheric Sciences, 2024. Available online: http://ww2010.atmos.uiuc.edu/(Gh)/guides/mtr/opt/mch/sct.rxml (accessed on 3 July 2024).
16. Andrews, L.C. *Field Guide to Atmospheric Optics*, 2nd ed.; SPIE Press: Bellingham, WA, USA, 2019. [CrossRef]
17. NOAA. The Color of Clouds. NOAA, JetStream, Topic Matrix, Clouds. Last Updated: 28 July 2023. Available online: https://www.noaa.gov/jetstream/clouds (accessed on 3 July 2024).
18. University of Twente. Atmospheric Scattering. University of Twente, ITC, Living Textbook. Available online: https://ltb.itc.utwente.nl/498/concept/81855 (accessed on 3 July 2024).
19. Kursun, O.; Dinc, S.; Favorov, O.V. Contextually Guided Convolutional Neural Networks for Learning Most Transferable Representations. In Proceedings of the 2022 IEEE International Symposium on Multimedia (ISM), Naples, Italy, 5–7 December 2022; pp. 210–213. [CrossRef]
20. Kursun, O.; Sarsekeyev, B.; Hasanzadeh, M.; Patooghy, A.; Favorov, O.V. Tactile Sensing with Contextually Guided CNNs: A Semisupervised Approach for Texture Classification. In Proceedings of the 2023 Seventh IEEE International Conference on Robotic Computing (IRC), Laguna Hills, CA, USA, 11–13 December 2023; pp. 25–30. [CrossRef]
21. Russell, R.; Kursun, O. Cloud Radiance HSI Dataset, 2024. IEEE Dataport 2024. [CrossRef]
22. Zhao, Z.; Zheng, P.; Xu, S.; Wu, X. Object detection with deep learning: A review. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 3212–3232. [CrossRef] [PubMed]
23. Iman, M.; Arabnia, H.R.; Rasheed, K. A review of deep transfer learning and recent advancements. *Technologies* **2023**, *11*, 40. [CrossRef]
24. Dosovitskiy, A.; Springenberg, J.T.; Riedmiller, M.; Brox, T. Discriminative unsupervised feature learning with convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 766–774.
25. Fernandez-Delgado, M.; Cernadas, E.; Barro, S.; Amorim, D. Do We Need Hundreds of Classifiers to Solve Real World Classification Problems? *J. Mach. Learn. Res.* **2014**, *15*, 3133–3181.
26. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
27. Alpaydin, E. *Introduction to Machine Learning*, 3rd ed.; The MIT Press: Cambridge, MA, USA, 2014.
28. Kursun, O.; Patooghy, A. An Embedded System for Collection and Real-Time Classification of a Tactile Dataset. *IEEE Access* **2020**, *8*, 97462–97473. [CrossRef]
29. Scikit-Learn. Ensembles: Gradient Boosting, Random Forests, Bagging, Voting, Stacking. Available online: https://scikit-learn.org/stable/modules/ensemble (accessed on 31 May 2024).
30. IBM. What Is Random Forest? Available online: https://www.ibm.com/topics/random-forest (accessed on 31 May 2024).
31. Yosinski, J.; Clune, J.; Bengio, Y.; Lipson, H. How transferable are features in deep neural networks? In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 3320–3328.
32. Scikit-Learn Contributors. sklearn.Metrics.Matthews_Corrcoef. Available online: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.matthews_corrcoef.html (accessed on 26 June 2024).