Deep Learning-Based Daytime Cloud Cover Estimation from All-Sky Camera Imaging: A Postprint

Authors: Che Lei, Li Leilei, Liu Liyong

Date: 2024-07-03T00:00:00+00:00

Abstract

Cloud cover is one of the important evaluation parameters for site selection of ground-based photoelectric telescopes in astronomy. To address the issues in daytime cloud cover calculation from all-sky camera imaging, a deep learning-based model for daytime cloud cover calculation from all-sky camera imaging is proposed. In the cloud detection layer, the model enhances both the memory capability for cloud features and the extraction capability of deep features by constructing a channel weighting-feature fusion (CWFF) structure to accomplish the cloud detection task; in the cloud cover calculation layer, the model proposes a cloud cover calculation method based on the cloud detection model, which effectively reduces the error rate of cloud cover calculation. Experimental results demonstrate that the proposed method achieves a comprehensive accuracy exceeding 95% in the cloud detection task, and a mean absolute error not exceeding 5% in the cloud cover calculation task.

Full Text

Preamble

ChinaXiv Vol. 42, No. 2

June 2024

PROGRESS IN ASTRONOMY Vol. 42, No. 2, June 2024
doi: 10.3969/j.issn.1000-8349.2024.02.11

Research on Daytime Cloudiness Calculation for All-sky Camera Imagery Based on Deep Learning

CHE Lei[1], LI Lei-lei[1], LIU Li-yong[2]
(1. School of Information Management, Beijing Information Science and Technology University, Beijing 100192, China;
2. National Astronomical Observatories, Chinese Academy of Sciences, Beijing 100101, China)

Abstract

Cloudiness is one of the critical evaluation parameters for ground-based photoelectric telescope site selection in astronomy. Addressing the challenges in daytime cloudiness calculation from all-sky camera imagery, this paper proposes a deep learning-based model for daytime cloudiness calculation. In the cloud detection layer, the model constructs a Channel Weighting-Feature Fusion (CWFF) structure to enhance cloud memory capacity and deep feature extraction for improved cloud detection performance. In the cloudiness calculation layer, the model introduces a novel method based on the cloud detection model that effectively reduces the error rate in cloudiness calculation. Experimental results demonstrate that the proposed method achieves comprehensive accuracy exceeding 95% in cloud detection tasks, with an average absolute error not exceeding 5% in cloudiness calculation tasks.

Key words: all-sky camera; cloudiness calculation; deep learning; U-Net

1 Introduction

Currently, cloudiness calculation primarily employs satellite detection and ground-based all-sky camera monitoring [1]. Satellite detection offers advantages in data accessibility, multi-band observation, and long-term data accumulation. However, satellite-based cloudiness calculation is limited by spatial and temporal resolution, where cloud cover over a typical site is represented by only a few pixels, and detection intervals are on the order of days or hours. In contrast, ground-based all-sky cameras provide stable performance and real-time monitoring, making them widely used for cloudiness calculation and evaluation at various observatory sites [2; 3].

Early meteorological cloudiness assessment relied on trained observers conducting visual evaluations. With advances in computer technology and big data processing, cloudiness calculation methods have gradually shifted from manual observation to automated and intelligent processing. Current approaches primarily include threshold-based methods, clustering methods, machine learning, and deep learning techniques.

In threshold-based methods, researchers have explored various threshold setting and optimization strategies to enhance applicability and accuracy. Huo and Lü [4] established correlations between pixel RGB values and spatial positions, using theoretical and empirical values to determine thresholds for preliminary cloud detection experiments. Qin et al. [5] exploited spectral differences between

clouds and snow in satellite imagery, employing Otsu's method for threshold-based cloud segmentation and extracting cloud shape features to successfully separate clouds from other ground objects. Ma et al. [6] developed a channel synthesis operation cloud detection method based on threshold techniques, significantly mitigating geographical location effects on cloud detection.

For clustering methods, researchers have introduced multi-dimensional features and multi-scale information to improve detection accuracy. Chen et al. [7] proposed a dual-threshold algorithm for cloud signal detection that effectively eliminates aerosol interference and accurately identifies cloud signals. Li et al. [8] investigated thin cloud identification and impact assessment algorithms, combining fuzzy C-means clustering to effectively recognize and analyze thin clouds in ground-based optical astronomical images, enabling automatic monitoring and real-time evaluation of cloud distribution and impact, thereby improving ground-based optical observation efficiency.

In machine learning approaches, researchers have employed traditional methods such as Support Vector Machine (SVM) and Backpropagation Neural Networks (BP neural networks) for cloud detection. Joshi et al. [9] proposed the STmask method based on SVM, which successfully improved classification performance across various land cover types. Gao et al. [10] introduced a dynamic threshold method based on normalized difference cloud index for preliminary cloud detection, and further developed a BP neural network-based algorithm that eliminated subjective threshold selection and demonstrated good performance on large-scale complex remote sensing imagery.

Deep learning represents algorithms that attempt to achieve high-level data abstraction using multiple processing layers with complex structures or multiple non-linear transformations. Several deep learning frameworks have been applied in computer vision, image segmentation, and natural language processing [11–14], achieving excellent results. In recent years, experts have conducted cloud detection research based on deep learning techniques. Kang and Sun [15] utilized artificial neural networks for automatic cloud detection, achieving semantic segmentation of clouds and sky with improved detection accuracy and generalization capability. Xia et al. [16] designed a convolutional neural network with four convolutional layers and one fully connected layer that effectively extracted basic cloud features, achieving segmentation accuracy above 85%. Pei et al. [17] employed a Fully Convolutional Network (FCN) model with multiple upsampling operations, improving cloud detection accuracy to 90.11% and essentially achieving end-to-end pixel-level detection capability.

Despite the advantages of deep learning in cloud detection, its application to all-sky camera imagery faces significant challenges due to cloud complexity and non-uniform light intensity, substantially affecting segmentation accuracy. As cloud complexity increases, maintaining segmentation accuracy becomes increasingly difficult. Moreover, existing semantic segmentation models demonstrate good accuracy only for specific scenes, showing insufficient generalization capability for all-sky camera imagery [18]. Additionally, uniform processing of

entire image features in convolutional operations reduces attention to important features. To address these issues, we propose a deep learning-based cloudiness calculation model for all-sky camera imagery, with three main contributions: (1) For cloud detection in all-sky camera imagery, we build upon the U-shaped network (U-Net) and propose a CWFF feature extraction structure to enhance deep semantic extraction, making segmentation results better reflect the characteristics of all-sky camera imagery. (2) We propose a cloudiness calculation method based on the cloud detection layer that directly calculates cloudiness results from feature matrices, effectively avoiding post-imaging calculation and improving efficiency. (3) We conduct comparative experiments on three typical datasets: conventional cloud distribution, all-sky cloud distribution, and light-interference scenarios, comparing fixed threshold, dynamic threshold, method from [16], FCN, U-Net, U-Net+Residual (denoted "+Residual"), U-Net+Attention (denoted "+Attention"), and our proposed method for both cloud detection and cloudiness calculation. Results demonstrate that our method outperforms others.

2 Model

The deep learning-based daytime cloudiness calculation model for all-sky camera imagery (shown in Figure 1 [Figure 1: see original paper]) comprises three layers. The first layer is the data processing layer, which performs preprocessing operations on raw data, including standardization and augmentation. The second layer is the cloud detection layer, which processes the preprocessed data through a U-Net model with CWFF feature extraction structure for cloud detection. The third layer is the cloudiness calculation layer, which applies threshold segmentation to the feature matrix from the cloud detection layer to determine cloud pixels in all-sky camera imagery, ultimately achieving cloudiness calculation.

2.1.1 Data Annotation

Since all-sky camera images vary in size and bit depth, uniform processing is required before training. First, cloud objects are cropped using an inscribed circle approach, ensuring the cloud object's boundaries are tangent to the image edges with a size of 256×256. Second, the bit depth is uniformly converted to 24-bit PNG format. Finally, LabelMe software is used to annotate cloud pixels in the imagery. During annotation, multiple closed curves mark cloud objects to ensure clouds and sky are separated by the curves, with cloud objects set to white and sky objects to black. The generated JSON files are converted to labeled images.

2.1.2 Data Augmentation

Appropriate data augmentation helps mitigate overfitting, improves model robustness and generalization, and addresses sample imbalance. During prepro-

cessing, random rotation, translation, and shearing transformations are applied to augment data diversity (see Figure 2 [Figure 2: see original paper]). The augmented images are then paired with their corresponding labels to generate image arrays. This preprocessing pipeline effectively improves data quality and quantity, enhancing model performance and robustness.

Figure 2 Data augmentation. Note: a) Original image; b) Flip-rotation-stretch; c) Rotation-translation; d) Shear-translation.

2.2 Cloud Detection Layer

Cloud detection is essentially a semantic segmentation task. U-Net is a classic semantic segmentation model consisting of two symmetric parts: an encoding network for capturing contextual information and a decoding network that mirrors the encoder to produce segmentation outputs. By fusing downsampling and upsampling features, U-Net obtains more accurate contextual information for improved segmentation. However, during forward propagation, the symmetric structure may cause information loss and varying channel correlations as convolutional layers deepen. Combining attention mechanisms [19; 20] and residual connections, we integrate CWFF feature extraction into U-Net to address the inability of lower convolutional layers to effectively focus on high-weight features and to mitigate model degradation. The method first assigns weights to feature channels through a channel weighting unit, then extracts target features while appropriately memorizing upper-layer features through a feature fusion unit. This yields more accurate and stable feature representations, enhancing model performance and efficiency. The CWFF-based feature extraction structure is shown in Figure 3 [Figure 3: see original paper].

Figure 3 CWFF-based feature extraction structure.

Each CWFF comprises a feature fusion unit and a channel weighting unit. Assuming each layer's input feature vector x_i has dimensions $[h, w, c]$, the channel weighting unit first applies global average pooling to compress it into a $[1, 1, c]$ feature vector F_{sq}:

$$F_{sq} = \frac{1}{h \times w} \sum_{k} u_k(i, j)$$

where u_k represents the k-th channel in the feature map, $k \in [1, c]$. A fully connected layer establishes channel correlations, outputting c weight coefficients s:

$$s = \sigma(W_2 \delta(W_{1F_{sq}}))$$

where $W_{1F_{sq}}$ represents the first fully connected operation with $W_1 \in \mathbb{R}^{c/r \times c}$, and r is a reduction factor to decrease channel count and computational cost.

The activation function δ maintains dimensionality. A second fully connected operation multiplies by $W_2 \in \mathbb{R}^{c/r \times c}$, followed by activation function σ to obtain weights s for assignment to initial feature maps.

The weighted feature vector F_w is input to the feature fusion unit, which performs three rounds of convolution and batch normalization to obtain feature vector $F(x_i)$. To prevent performance degradation when learned features approach zero, the model adds the input x_i to $F(x_i)$ for the final output x_{i+1}:

$$x_{i+1} = f(H(x_i) + F(x_i))$$

where x_i and x_{i+1} denote the input and output of the i-th feature fusion unit, F represents learned features, and f is the δ activation function. $H(x_i) = x_i$ denotes identity mapping when learned features are zero. This ensures more direct gradient flow and better adaptation to deep network training. This completes one full CWFF extraction process. The U-Net model incorporating CWFF is shown in Figure 4 [Figure 4: see original paper].

Figure 4 U-Net model with CWFF feature extraction structure.

2.3 Cloudiness Calculation Layer

After feature extraction by the cloud detection layer, the final activation function generates a feature matrix:

$$A_{m\times n} = \begin{pmatrix} a_{m-1,0} & \cdots & a_{m-1,n-1} \\ \vdots & \ddots & \vdots \\ a_{0,0} & \cdots & a_{0,n-1} \end{pmatrix}$$

where $m, n \in [0, 255]$, and each $a_{i,j}$ consists of two components (x, y), with x representing the probability of sky background and y representing cloud probability, where $i \in [0, m-1]$, $j \in [0, n-1]$, and $x, y \in [0, 1]$.

The cloudiness calculation layer first applies threshold segmentation as shown in Equation (5). The segmentation threshold τ is set to 0.9 (we initially used 0.5 with step size 0.05 up to 0.95, finding $\tau = 0.9$ yields cloudiness statistics closest to standard values). For $a_{i,j}$ where $y > \tau$, it is set to $(0, 1)$; for $y \leq \tau$, it is set to $(1, 0)$:

$$a_{i,j} = \begin{cases} (0,1), & y > \tau \\ (1,0), & y \leq \tau \end{cases}$$

Pixel classification then introduces matrix $B_{1\times 2} = (0 \quad 1)$, which multiplies $a_{i,j}$ to determine whether each pixel belongs to cloud or sky. A result of 0 indicates sky pixel, while 1 indicates cloud pixel:

The processed feature matrix becomes:

$$A_{m\times n} = \{ea_{i,j}\} = \{a_{i,j} \times B\} = \begin{pmatrix} ea_{0,0} & \cdots & ea_{m-1,0} \\ \vdots & \ddots & \vdots \\ ea_{0,n-1} & \cdots & ea_{m-1,n-1} \end{pmatrix}$$

where elements are binary vectors (0 for sky, 1 for cloud).

During preprocessing, training labels were converted to square images of size $m \times n$ containing all cloud pixels within the inscribed circle. The cloud feature matrix generated by the detection model is also an $m \times n$ square. Therefore, cloudiness is calculated as the percentage of cloud pixels within the inscribed circle of the square matrix, as shown in Equation (8). Summing all cloud pixel values and dividing by the total number of pixels in the inscribed circle yields the final cloudiness result:

$$\text{Cloudiness} = \frac{\sum ea_{i,j}}{\sum_{i,j} \mathbb{I}(\text{pixel in inscribed circle})}$$

2.4 Loss Function

The loss function measures the difference between predictions and ground truth during neural network training, with its magnitude reflecting prediction accuracy. Cross-entropy, a commonly used loss function, effectively addresses the learning rate reduction problem of mean squared error in gradient descent and evaluates similarity between training samples. Since cloud detection in our cloudiness calculation is essentially a binary classification problem, we employ binary cross-entropy as the loss function:

$$\text{loss} = -\frac{1}{N}\sum_{i=1}^{N}[x_i \log y_i + (1 - x_i)\log(1 - y_i)]$$

where x_i is the ground truth and y_i is the predicted value.

3 Experiments

3.1 Experimental Environment and Dataset

The experimental environment is detailed in Table 1 . The dataset comprises 1,500 images captured by the same all-sky camera, including images with the sun at various altitudes with partial removal of ground buildings. The dataset was randomly split into 1,200 training images and 300 test images, with 150 training epochs, batch size of 8, Adam optimizer, initial learning rate of 0.0001, and learning rate reduction factor of 0.5.

Table 1 Experimental environment

Component	Specification
Python	TensorFlow, Keras
OS	Windows 10 IoT Enterprise
CPU	Intel Core i7 9xx (Nehalem Class Core i7) 2.40 GHz
RAM	16.0 GB
IDE	PyCharm

3.2.1 Evaluation Metrics

Cloud detection, as the foundation of cloudiness calculation, is an image segmentation problem. Its confusion matrix is shown in Table 2 , where TP represents cloud pixels predicted as cloud, TN represents sky pixels predicted as sky, FP represents sky pixels predicted as cloud, and FN represents cloud pixels predicted as sky.

Table 2 Confusion matrix

	Predicted Cloud	Predicted Sky
Actual Cloud	TP	FN
Actual Sky	FP	TN

From the confusion matrix, we derive Pixel Accuracy (PA), Intersection over Union (IOU), and Mean Intersection over Union (MIOU) metrics:

Pixel Accuracy (PA):

$$\mathrm{PA} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

Intersection over Union (IOU):

$$\mathrm{IOU} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}$$

Mean Intersection over Union (MIOU):

$$\mathrm{MIOU} = \frac{1}{k+1} \sum_{i=0}^{k} \mathrm{IOU}_i$$

Experimental comparison baselines include: (1) Fixed threshold, which determines data boundaries to evaluate conformance; (2) Dynamic threshold, which calculates region-specific thresholds; (3) Method from [16], a three-layer convolutional neural network; (4) FCN, which replaces fully connected layers with convolutional layers and uses upsampling to restore original dimensions; (5)

Original U-Net; (6) +Residual, adding residual blocks to U-Net; (7) +Attention, adding attention mechanisms to U-Net; (8) Our proposed method.

Cloud detection comparisons only include deep learning methods (3)–(8). Fixed and dynamic thresholds are cloudiness calculation methods used in Section 3.3 experiments.

Table 3 presents cloud detection metric comparisons. Compared with [16], FCN, U-Net, +Residual, and +Attention, our method relatively improves PA by 6.45%, 2.29%, 1.19%, 0.79%, and 0.87%; IOU by 12.28%, 4.53%, 2.34%, 1.55%, and 1.7%; and MIOU by 12.28%, 4.51%, 2.33%, 1.51%, and 1.68%. These results demonstrate our method's superiority.

Table 3 Cloud detection metric comparison

Method	PA	IOU	MIOU
[16]	0.9839	-	-
U-Net	-	-	-
+Residual	0.9831	-	-
+Attention	-	-	-
Ours	-	-	-

Figure 5 [Figure 5: see original paper] shows loss convergence during training. Method [16]'s loss stops decreasing around 0.2. FCN, U-Net, +Residual, and +Attention, as deep networks, show similar convergence, though +Residual and +Attention perform slightly better than FCN and U-Net. Our method achieves the best convergence.

Figure 5 Loss function convergence comparison

3.3.1 Global Data Cloudiness Calculation Comparison

To evaluate our model's effectiveness, we input 300 test images into the cloud detection models listed in Table 4 . Among deep learning methods, [16] shows lower accuracy due to simple feature extraction. Without attention or residual mechanisms, FCN slightly outperforms U-Net. Our proposed method achieves state-of-the-art accuracy of 3.53, confirming its effectiveness.

Table 4 Mean absolute difference from expert annotation for all test sets

Method	Mean Absolute Difference
Fixed threshold	-
Dynamic threshold	-
[16]	-
FCN	-
U-Net	-

Method	Mean Absolute Difference
+Residual	-
+Attention	-
Ours	3.53

3.3.2 Typical Scenario Analysis

To further validate our method, we analyze results from three typical scenarios in the 300 test images: conventional cloud distribution, all-sky cloud distribution, and light-interference conditions. Conventional distribution features clear cloud-sky boundaries; all-sky distribution involves overcast conditions easily confused with haze; light-interference scenarios involve solar illumination behind clouds creating blind spots prone to misjudgment. Figures 6–8 [FIGURE:6–8] and Tables 5–7 [TABLE:5–7] present detection results and cloudiness values S (calculated via Equation (8)), pixel accuracy PA (via Equation (10)), and absolute differences from expert annotation. Note that PA measures per-pixel prediction correctness, which does not fully correlate with cloudiness difference from ground truth.

Figure 6 Cloud detection results for conventional distribution

Figure 7 Cloud detection results for all-sky distribution

Figure 8 Cloud detection results for light-interference conditions

Table 5 Cloudiness calculation results for conventional distribution

Method	Result S (%)	Absolute Difference	PA (%)
Expert annotation	-	-	-
Fixed threshold	-	-	-
Dynamic threshold	-	-	-
[16]	-	-	-
FCN	-	-	-
U-Net	-	-	-
+Residual	-	-	-
+Attention	-	-	-
Ours	-	-	-

Table 6 Cloudiness calculation results for all-sky distribution

Method	Result S (%)	Absolute Difference	PA (%)
Expert annotation	-	-	-
Fixed threshold	-	-	-
Dynamic threshold	-	-	-

Method	Result S (%)	Absolute Difference	PA (%)
[16]	-	-	-
FCN	-	-	-
U-Net	-	-	-
+Residual	-	-	-
+Attention	-	-	-
Ours	-	-	-

Table 7 Cloudiness calculation results for light-interference conditions

Method	Result S (%)	Absolute Difference	PA (%)
Expert annotation	-	-	-
Fixed threshold	-	-	-
Dynamic threshold	-	-	-
[16]	-	-	-
FCN	-	-	-
U-Net	-	-	-
+Residual	-	-	-
+Attention	-	-	-
Ours	-	-	-

Figure 6 and Table 5 compare methods under conventional distribution. Deep learning methods clearly outperform traditional approaches. While [16]'s simple three-layer convolution weakens deep semantic extraction, multi-layer convolutional and feature fusion networks (FCN and U-Net) better restore fundamental characteristics of all-sky imagery. Our method provides more detailed representation, achieving the highest absolute difference and PA.

Figure 7 and Table 6 compare methods under all-sky distribution. Traditional and deep learning methods show similar performance in overcast conditions. In Figure 7, boxes 1–5 mark white buildings misidentified as clouds by fixed threshold, dynamic threshold, [16], and +Attention, while FCN, U-Net, +Residual, and our method correctly identify them. FCN achieves PA closest to expert annotation at 98.13% with smallest difference (0.39), followed by our method at 98.01% and 0.52.

Solar illumination behind thin clouds creating large blind spots is common and challenging. Figure 8 and Table 7 compare methods under light interference. Fixed threshold, dynamic threshold, and [16] show significantly reduced PA. Deep networks still perform excellently; while +Residual achieves smallest absolute difference (0.07), its PA (94.56%) is lower than our method's 95.93%, possibly because +Residual misidentifies some sky background or solar glare as clouds.

3.4 Conclusion

This paper proposes a deep learning-based daytime cloudiness calculation model for all-sky camera imagery. The CWFF feature extraction structure enhances representation learning and detection performance. Experimental results show our model achieves PA of 0.9917, IOU of 0.9836, and MIOU of 0.6557 under normal conditions. The proposed cloudiness calculation method based on detection models achieves an average absolute difference of 3.53, representing significant improvement over baseline and other semantic segmentation models. While effective for daytime cloudiness, nighttime identification remains challenging due to insufficient lighting and unclear cloud edges. Future work will explore transferring the trained daytime model to nighttime scenarios using transfer learning.

References

[1] Tao F, Hu S. Meteorological Hydrological and Marine Instruments, 2017, 34(04): 1
[2] Skidmore W, Schock M, Sagnier E, et al. GAT, 2008: 862
[3] Martinis C, Wilson J, Zablowski P, et al. PASP, 2013, 125(923): 56
[4] Huo J, Lü D. Journal of Nanjing Institute of Meteorology, 2002, 242: 246
[5] Qin Y, Fu Z, Zhou F, et al. Geomatics and Information Science of Wuhan University, 2014, 39(02): 234
[6] Ma F, Zhang Q, Guo N, et al. Chinese Journal of Atmospheric Sciences, 2007, 119: 128
[7] Chen S, Chang J, Liu Z, et al. Chinese Journal of Lasers, 2022, 49(11): 154
[8] Li X, Cai H, Li H, et al. Laser & Optoelectronics Progress, 2022, 59(16): 171
[9] Joshi P P, Wynne R H, Thomas V A. IJAEOG, 2019, 82: 101898
[10] Gao J, Wang K, Tian X, et al. Journal of Infrared and Millimeter Waves, 2018, 37(04): 477
[11] Chen Y, Huang Y, Zhang J. Computer Engineering, 2023, 49(6): 257
[12] Li X, Zhang Q, Song C, et al. Computer Engineering, 2023, 49(5): 1
[13] Li S, Geng L, Wang P. Computer Engineering, 2023, 49(4): 272
[14] Wang C, Sun J, Yang W. Computer Engineering, 2023, 49(2): 37
[15] Kang X, Sun L. Journal of PLA University of Science and Technology (Natural Science Edition), 2005, 102: 106
[16] Xia M, Shen M, Wang J, et al. Journal of System Simulation, 2018, 1623: 1630
[17] Pei L, Liu Y, Tan H, et al. Laser & Optoelectronics Progress, 2019, 56(05): 226
[18] Tian J F, Ge L, Wu Y, et al. PASP, 2022, 134(1033): 035002
[19] He K, Zhang X, Ren S, et al. IEEE, 2016, 770: 778
[20] Hu J, Shen L, Sun G. IEEE, 2018, 7132: 7141

Note: Figure translations are in progress. See original paper for figures.