International Conference on Astronautics and Space Exploration (iCASE)
Hsinchu, Taiwan, November 12-14, 2019

A Real-Time Cloud Detection System Using an End-to-End Multiscale Deep Learning Approach

CHIN-SHYURNG FAHN AND MENG-LUEN WU
Department of Computer Science and Information Engineering
National Taiwan University of Science and Technology
No. 43, Keelung Rd., Sec. 4, Da'an Dist., Taipei City 10607, Taiwan (R.O.C.)
csfahn@mail.ntust.edu.tw, d10015015@mail.ntust.edu.tw

JER LING AND MING-YUAN YEH
System Engineering Division, National Space Organization
National Applied Research Laboratories
No. 9, Zhanye 1st Rd., East Dist., Hsinchu City 30078, Taiwan (R.O.C.)
se01@nspo.narl.org.tw, marco@nspo.narl.org.tw

ABSTRACT

In remote sensing, photos containing many clouds are out of value and will be discarded. In satellite imagery, cloud detection is required to reduce the satellite camera power spent on shooting unwanted cloud regions. Besides high resolution cameras that consume much power on imaging, low resolution cameras with microcomputer systems may be also power-consuming if they detect cloud regions appearing in satellite images inefficiently. In the working environment of a satellite, some issues are worth investigating. First, because satellites are fast flying vehicles, the detection time must be short. Second, because the power of a satellite is extremely limited, the power consumption must be economical. Third, the flying direction of a satellite cannot be rolled back in every day, the accuracy of the cloud detector must be high enough. Forth, cloud regions present in multiple scales, so the detector must be able to find clouds in different scales. Accordingly, in this paper, we adopt a multiscale deep learning scheme to accomplish a real-time, low power consumption, and high accurate cloud detector.

In our proposed detector, we use a convolutional neural network that applies skip-connection to fetching the residual values of previous layers to overcome the problem of gradient disappearance or explosion. The end-to-end multiscale deep learning approach is employed, in which both the coordinates and sizes of potential clouds act as the input to train the network. Consequently, no sliding window is required and all

potential clouds can be detected in a single iteration. Experimental results reveal that our proposed approach can achieve the real-time detection requirement, and the true positive rate of cloud detection is more than 90% for both the inside and outside testing, which is suitable for the actual deployment of cloud detection on a satellite system.

***Keywords**: Cloud detection, convolutional neural network, deep learning, residual network, multiscale detection, end-to-end detection.*

1. INTRODUCTION

Satellite images are widely used in land telemetry, disaster prevention, disaster relief, and commercial services. These images are shot by high resolution cameras equipped on a satellite. In Fig. 1, the land is shrouded with lots of clouds in a satellite image. Such a content may make the image be valueless, which will be abandoned or deleted eventually. As a result, if the camera system can be aware of too many clouds existing in the field of view (FOV), the power for capturing this kind of high resolution images can be saved.

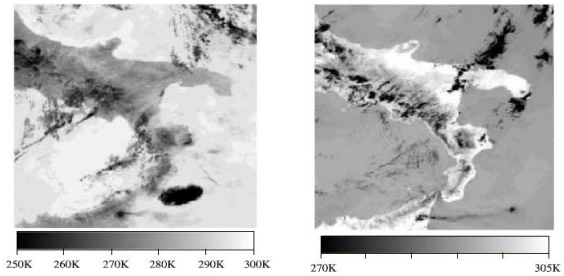
Fig. 1. A satellite image of many clouds shrouding the land.

Because the satellite is a fast flying vehicle, the time of cloud detection must be short; otherwise, the detected result cannot match the real situation. Clouds commonly appear in various shapes and sizes, so the detection system must be capable of handling all cases. To achieve this, we propose a machine learning approach to detect clouds. In the past, the cloud detection method is composed of multiple steps, such as image pre-processing, feature extraction, and cloud classification. However, both the image pre-processing and feature extraction spend much computation time during the detection phase. To surmount this problem, our method adopts an end-to-end multiscale deep learning approach that the satellite image is directly sent into the neural network, and the coordinates of clouds are returned without any post-processing steps. In the end-to-end scheme, we exploit a machine learning method that immediately train the whole

satellite image with marked cloud coordinates. In this manner, our method is purely based on machine learning, and other steps are not required.

2. RELATED WORK

In the area of cloud detection, there are many distinct methods used for different kinds of images. According to thermal infrared information, in 2004, Pergola et al. proposed a method to detect volcanic ash clouds from AVHRR bands captured by the camera system equipped on a satellite [1]. Figure 2 shows some sample images in pseudo colors that represent different ranges of elevation.



(a) (b)

Fig. 2. Infared cloud images in pseudo colors representing different ranges of elevation: (a) 250K~300K; (b) 270K~305K.

Based on IKONOS images as shown in Fig. 3, in 2006, Lu et al. proposed a method to detect clouds and hazes [2]. In this kind of images, there are two specifications that consist of four multispectral bands, including blue, green, red, and near-infrared wavelengths, with four-meter spatial resolution, and one panchromatic band with one-meter spatial resolution.

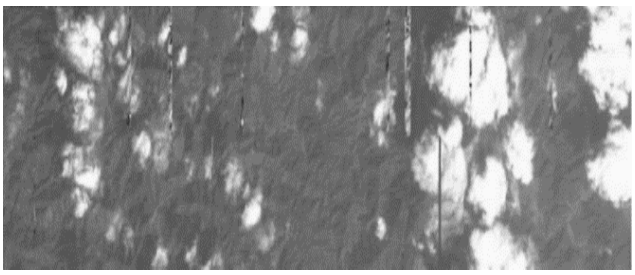
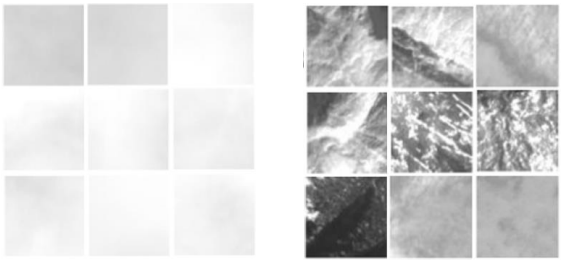


Fig. 3. A sample image captured from the IKONOS satellite.

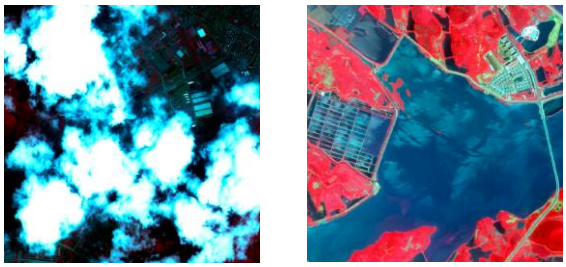
Using the reflection spectrum of ground objects, in 2015, Li et al. proposed a method to find thick clouds in satellite images [3]. In their method, the image is first divided into multiple small blocks as Fig. 4 shows, then SVMs are applied to detecting clouds within each block.



(a) (b)

Fig. 4. Cloud and ground object samples: (a) clouds; (b) ground.

In 2016, Bao et al. proposed a cloud detection method using the Gaofen dataset that is composed of single-band panchromatic images and four-band multispectral images, as Fig. 5 demonstrates [4].



(a) (b)

Fig. 5. Sample images chosen from the Gaofen dataset: (a) clouds; (b) water.

Some cloud detection methods have been developed to test on optical remote sensing images which are in the RGB color space. In 2018, Deng et al. adopted Simple Linear Iterative Clustering (SLIC) and Gabor features to detect clouds [5]. From a satellite image, they extracted cloud texture features for an SVM classifier. The main steps of their method are graphically shown in Fig. 6.

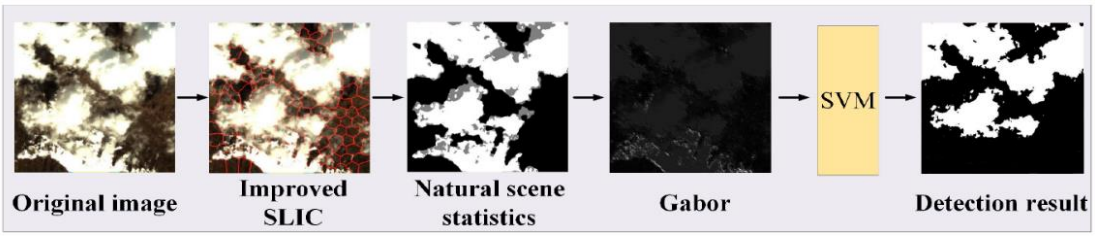


Fig. 6. Illustration of Deng's cloud detection method using improved SLIC and Gabor filter.

Based on optical remote sensing images, some existing cloud detection methods require object segmentation and needs classification on each segmented region, which is time-consuming and not suitable for the requirement of real-time cloud detection. On the other hand, some state-of-the-art methods without object segmentation need dedicated equipment. To solve these issues, in this paper, we propose an end-to-end

multiscale deep learning approach that only a convolutional neural network (CNN) architecture is required, and image processing steps are not necessary. In consequence, both the computational time and system performance are greatly improved.

3. END-TO-END CLOUD DETECTION

To achieve real-time and high performance detection, we use a deep learning scheme for an end-to-end cloud detection. Unlike traditional methods that use sliding windows to scan all possible regions separately, our proposed approach only reads the entire input image once, and the coordinates of the boxes bounding cloud regions are obtained in a single iteration. This approach is based on a CNN [6], which extracts the cloud image features automatically through multiple convolutional layers. Figure 7 shows the system diagram of our real-time cloud detection strategy using an end-to-end multiscale deep learning approach.

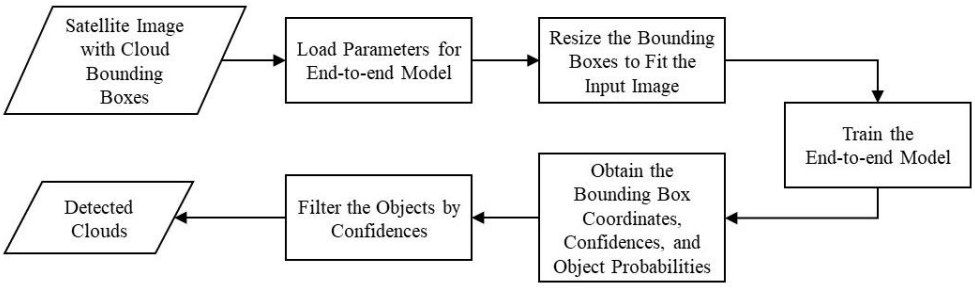


Fig. 7. The system diagram of our real-time cloud detection strategy using an end-to-end multiscale deep learning approach.

In a traditional CNN architecture, there are pairs of a convolutional layer and a pooling layer. These pairs of layers perform automatic image feature extraction and dimension reduction. Therefore, the image features are acquired from a training process. Subsequently, the fully connected layer, as known as a feed forward neural network, is used to classify the input, and the result is obtained through the neurons in its output layer. However, the fully connected layers have some issues. First, overfitting may occur when all neurons must be connected with each neuron in the next layer. Second, because of fully connectivity, the computational time is very long. Consequently, in the recent design of a CNN architecture, the fully connected layer is removed, and only convolutional and pooling layers remain in the network. In this paper, we also discard the fully connected layer, and the conception of the CNN architecture used for cloud detection is briefly demonstrated in Fig. 8.

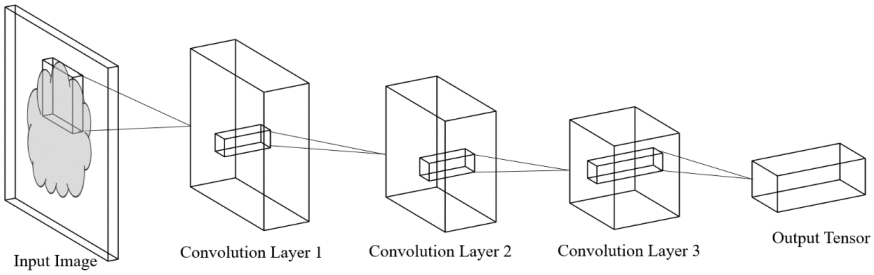
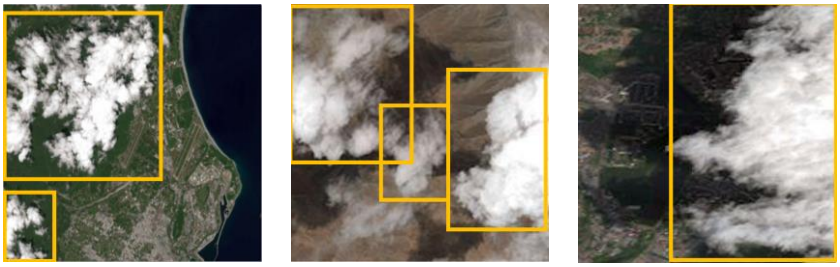


Fig. 8. The conception of the CNN architecture used for cloud detection.

In the past, to detect clouds, we must design a sliding window or an object proposal to focus on certain areas and create a classifier to check whether the detected region is a cloud or not. However, either the sliding window or object proposal spends much execution time because a complete detection process requires multiple iterations. Therefore, we develop an end-to-end multiscale deep learning approach to attain a faster and higher performance detection.

In the end-to-end approach, the training method is different from traditional methods. In particular, the whole satellite image, rather than only cloud regions, is used for training [7]. As shown in Fig. 9, in the training phase, we label a cloud region with the coordinates of its bounded box, including the starting position, say the lower left corner, and its width and height. These coordinate information act as the input together with the training image. The goal of our approach is to use the CNN architecture alone to detect clouds, without image pre-processing steps and not any sliding window or object proposal required.



(a) (b) (c)

Fig. 9. Samples of cloud regions labelled by bounding boxes in satellite images: (a) two separate bounding boxes; (b) two bounding boxes overlapped and one bounding box concatenated; (c) a single bounding box.

The training image is partitioned into $S \times S$ grid cells, each of which contains multiple bounding boxes and is responsible for calculating the probabilities of objects detected within the cell. Figure 10 shows the essentials of $S \times S$ grid cells of the output tensor in the CNN architecture.

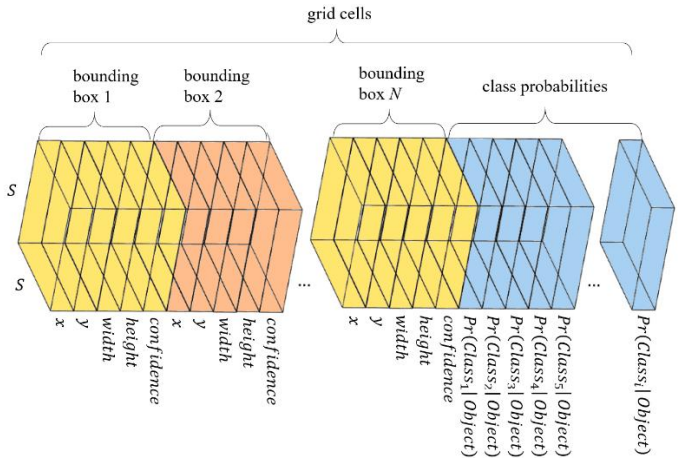


Fig. 10. Essentials of $S \times S$ grid cells of the output tensor in the CNN architecture.

To begin with, multiple bounding boxes are set up in a training image, which may across multiple grid cells. Next, reserve the bounding box whose confidence score of an object exceeds a threshold and mark it as a detected object. The following defines the confidence score of a bounding box to which an object pertains:

$$Conf(Object) \cong Pr(Object) \times IoU^{truth}_{prediction} \tag{1}$$

with $IoU^{truth}_{prediction} = \frac{Area\ of\ intersection}{Area\ of\ union}$

where $IoU^{truth}_{prediction}$ is the ratio of intersection of the prediction result and ground truth to the union of the prediction result and ground truth. The conception of how to calculate this ratio is illustrated in Fig. 11.

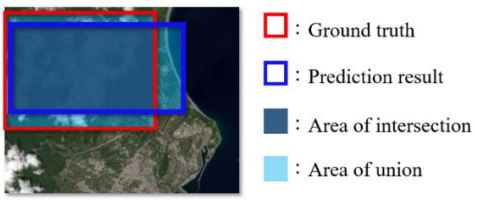


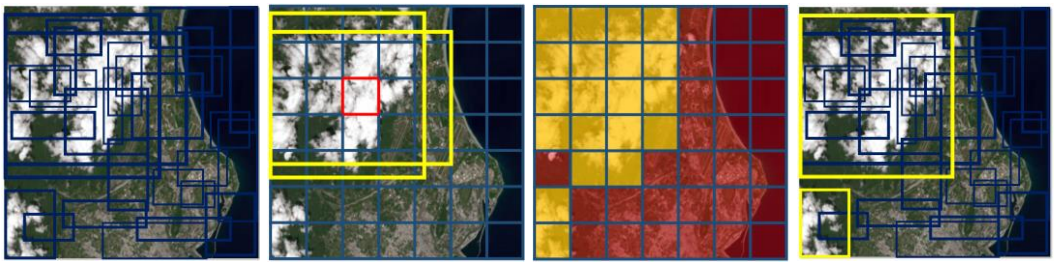
Fig. 11. The conception of calculating Intersection of Union.

Finally, for each grid cell, the confidence score of a specific class is calculated by the following equation.

$$\begin{aligned} Conf(Class_i) &= Pr(Class_i|\,Object) \times Conf(Object) \\ &\cong Pr(Class_i|\,Object) \times Pr(Object) \times IoU^{truth}_{prediction} \end{aligned} \tag{2}$$

In the detection phase, for a satellite image, the process of detecting clouds is carried out, as Fig. 12 demonstrates. First, Fig. 12(a) shows many bounding boxes randomly scattered in a satellite image, and the confidence score of the bounding box

surrounding an object is calculated. Then in Fig. 12(b), the bounding box whose confidence score higher than a threshold is marked in yellow. Subsequently, the confidence score of a specific class for each grid cell is also evaluated, and high confidence grid cells are marked in orange as shown in Fig. 12(c). At last, by aggregating the results, the cloud regions are detected and shown in Fig. 12(d).



(a) (b) (c) (d)

Fig. 12. Illustration of the detection phase: (a) bounding boxes randomly scattered in a satellite image; (b) a grid cell in red within high confidence bounding boxes marked in yellow; (c) high confidence grid cells marked in orange; (d) detected cloud regions marked in yellow.

The confidence scores of objects are obtained from training the CNN. This network calculates the confidence scores by the minimization of a loss function which serves as the core of the object detection method. After minimizing the loss function, both the coordinates and classes of detected objects can be similar to or the same as those of the objects cropped from the training set. The loss function is stated below [7]:

$$\begin{aligned} LOSS = {} & \lambda_{coordinate} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{I}_{ij}^{obj} \left[(b_{xi} - \hat{b}_{xi})^2 + (b_{yi} - \hat{b}_{yi})^2 \right] \\ & + \lambda_{coordindate} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{I}_{ij}^{obj} \left[\left(\sqrt{b_{wi}} - \sqrt{\hat{b}_{wi}} \right)^2 + \left(\sqrt{b_{hi}} - \sqrt{\hat{b}_{hi}} \right)^2 \right] \\ & + \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{I}_{ij}^{obj} (C_i - \hat{C}_i) + \lambda_{noObj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{I}_{ij}^{noObj} (C_i - \hat{C}_i) \\ & + \sum_{c \in classes} \mathbb{I}_{i}^{obj} (p_i(c) - \hat{p}_i(c)) \end{aligned} \tag{3}$$

where we empirically set $S = 7$, $\lambda_{coordindate} = 5$, and $\lambda_{noObj} = 0.5$. Without appropriate setting of $\lambda_{coordindate}$ and λ_{noObj}, there would be no objects detected. In the above equation, the first three terms have relations with the position, size, and class confidence, respectively. $\mathbb{I}_{ij}^{obj}$ acts as an object predictor, where i is the index of a grid cell containing multiple predictors, and j is the index of the bounding box used as a predictor. Particularly, the latter two terms in Eq. (3) are used for increasing the penalties of miss detection of objects, and decreasing the penalties for non-object

regions. Only the bounding boxes with the maximum $IoU_{prediction}^{truth}$ are predictors, and there are B predictors totally. In other words, bounding boxes that do not belong to the predictors are ignored. Conversely, $\mathbb{I}_{ij}^{obj}$ plays a non-object detector. In such a case, all the grid cells with confidence scores larger than zero are regarded as errors. Here, in a grid cell, we only choose the bounding box with the largest class confidence C for calculation. $\mathbb{I}_{i}^{obj}$ denotes a binary value whether a grid cell i contains the center of an object; for example, it is assigned to 1 when the grid cell i contains the center of an object; otherwise, it is set to 0. As a result, in the loss function, we only consider the grid cell containing the center of an object, rather than using all the grid cells that merely contain any part of an object.

Our CNN architecture is set up as shown in Fig. 13, which is composed of multiple convolutional layers to extract cloud image features. In the first layer, the network processes common contents of an image, followed by multiple layers to deal with the objects of three different sizes.

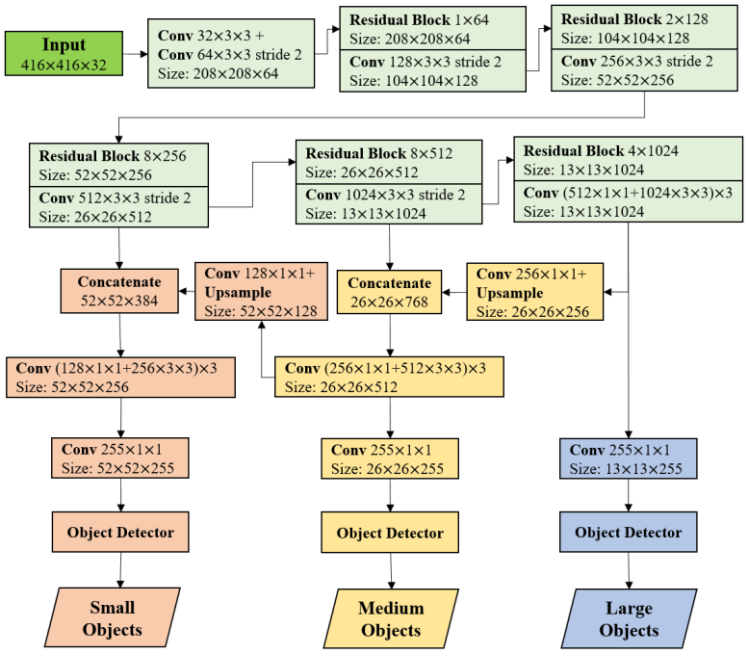


Fig. 13. The CNN architecture for our real-time cloud detection system.

Currently, the CNN is based on deep learning scheme that comprises more than 5 layers. However, the network consisting of excessive layers will result in gradient disappearance or gradient explosion [8], making it impossible to train the model correctly. Accordingly, the skip connection technique is employed to overcome such a problem, and the residual layer serves as a solution shown in Fig. 14 where the leaky ReLU function is selected as an activation function [7]. Traditionally, the ReLU

function is set to zero when an input value is less than or equal to 0, which may discard some subtle but decisive information. To surmount this problem, the leaky ReLU function is redefined as $0.1x$ when the input value x is less than or equal to 0.

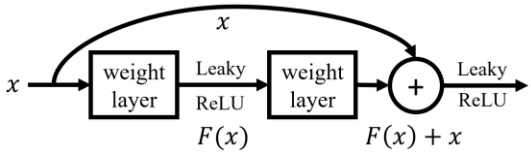


Fig. 14. Illustration of the residual learning method.

In addition to the skip connection technique, we use the concatenation technique between two layers to detect clouds in multiple scales [9]. However, concatenating two layers in different sizes is inconceivable, so we apply the upsampling technique to resizing the layers with lower dimensions [10]. Here, we adopt the nearest neighboring method as Fig. 15 illustrates.

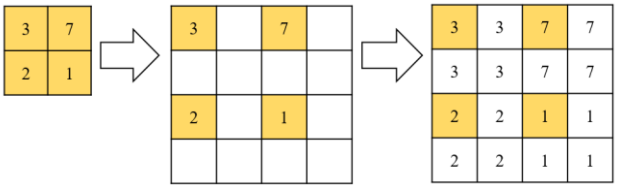


Fig. 15. The upsampling technique realized by the nearest neighboring method.

In the CNN architecture, we also split the layer with a higher dimension to some layers with lower dimensions. The splitting method is graphically shown in Fig. 16. In consequence, the layers with higher dimensions can be resized, and are assigned to multiple scales and layers.

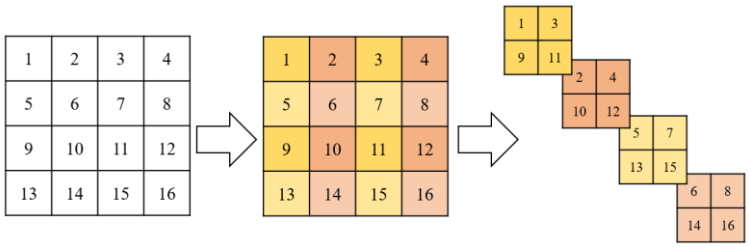


Fig. 16. Layer dimension reduction by the splitting method.

4. EXPERIMENTAL RESULTS

Many experiments are made for verifying the effectiveness of our developed real-time cloud detection system. In the experiments, we collect 800 satellite images either

with or without clouds. The clouds are in different sizes, thickness, and shapes, even appearing in arbitrary areas. Besides, there are various types of terrain captured in satellite images, some of which are as shown in Fig. 17.

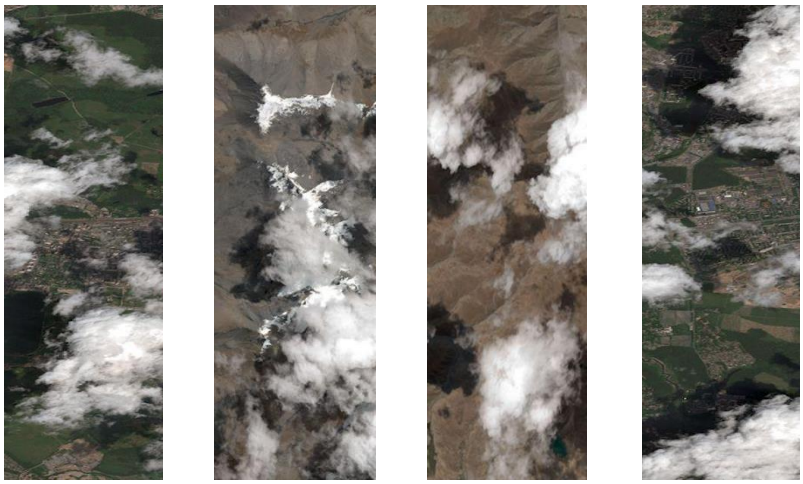
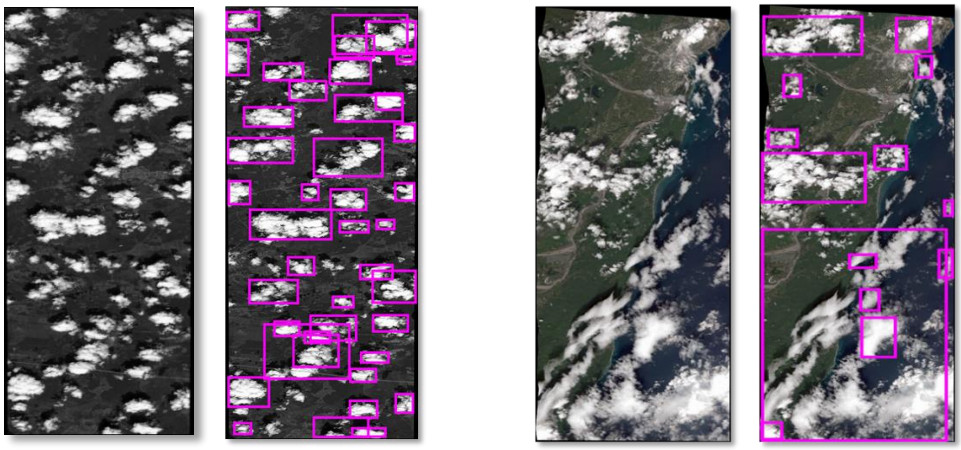


Fig. 17. Some samples of satellite images containing clouds shrouding terrain.

Figure 18 shows some results of cloud detection presented in the satellite images. By virtue of the CNN architecture, the detected clouds are marked in bounding boxes. As seen in the figure, most of large sized cloud regions are detected, and only too small clouds are missed.



(a) (b)

Fig. 18. Cloud detection results from different types of satellite images containing: (a) small clouds; (b) clouds with different sizes.

Herein, we use a statistical approach to evaluate the accuracy of the proposed method. Intersection of Union (IoU) is a measurement to determine the accuracy of detection results with the aid of bounding boxes. The value of IoU is calculated by

counting the overlapping pixels between the bounding boxes of detected clouds and the ground truth. There are two types of false detections defined on a pixel basis as Fig. 19 conceptually shows. One is false negative, which means the cloud pixel is not in any of the bounding boxes; the other is the false positive, which means the non-cloud pixel is within at least one bounding box.

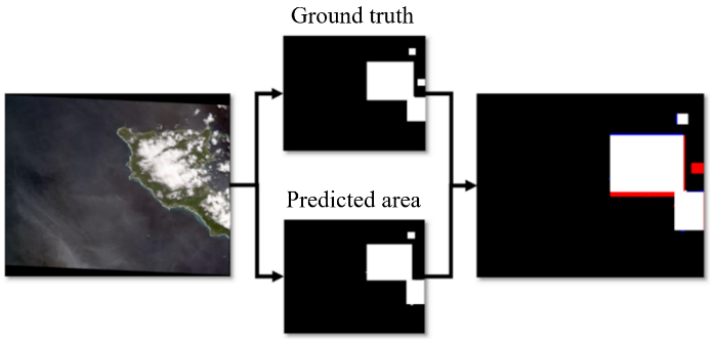


Fig. 19. An example of calculating IoU by the aid of: white regions indicating correctly detected bounding box, red regions indicating false negative, and blue regions indicating false positive.

The purpose of our developed system is to detect whether clouds shroud the land in a satellite image; hence, we also formulate such evaluation on an image basis, which means whether our developed system can correctly determine to shot the satellite image or not. In Fig. 20, we show the four types of cloud detection results for each image. The upper-left image is the true positive case that most bounding boxes are actually clouds. The upper-right image is the false negative case that most cloud regions are not within the bounding boxes. The lower-left image is the false positive case that most of the bounding boxes contain no clouds. The lower-right image is the true negative case that there are no bounding boxes and there exist actually no clouds in the image.

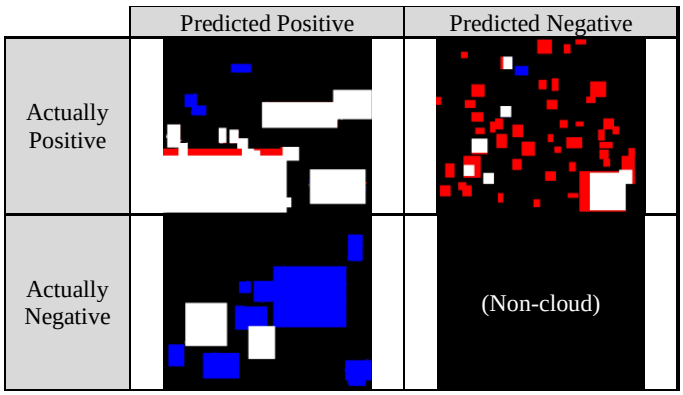


Fig. 20. Confusion matrix of four types of cloud detection results.

In the experimental result, the IoU acts as a prior criterion for accuracy evaluation. An image is regarded as correctly classified when the associated IoU is larger than a given threshold. By definition, the inside testing means the test set is the same as the training set, and the outside testing means an additional test set is used for testing. As listed in Table 1, when using the strictest threshold, say 90%, for the inside testing, the accuracy of cloud detection is 75%, but when descending the threshold to 60%, the accuracy of cloud detection raises to 87%. Nevertheless, for the outside testing, the accuracy of cloud detection is more or less than 86%, no matter what the threshold is set from 60% to 90% as Table 2 reveals.

Table 1 Accuracy of Cloud Detection for Inside Testing on the Image Basis

IoU	Correct	Total	Accuracy
≥60%	247	285	87%
≥70%	241	285	85%
≥80%	234	285	82%
≥90%	215	285	75%

Table 2 Accuracy of Cloud Detection for Outside Testing on the Image Basis

IoU	Correct	Total	Accuracy
≥60%	177	203	87%
≥70%	176	203	87%
≥80%	175	203	86%
≥90%	170	203	84%

Actually, when lots of clouds appear in the FOV, the camera system may abandon the captured image. Therefore, another evaluation method is that when there exist clouds in an image, we set the image as positive; otherwise, set as negative. We use both the true positive rate (TPR) and positive predictive value (PPV) to evaluate the performance, and work out as follows.

$$TPR = \frac{TP}{TP + FN} \tag{4}$$

$$PPV = \frac{TP}{TP + FP} \tag{5}$$

The experimental results are shown in Table 3 and Table 4 for the inside and outside testing, respectively. From these tables, we can see that the TPRs are 100% in both the inside testing and outside testing. It means our developed system will correctly alert when clouds exist. The only mistake made by our developed system is a false alarm. That is, the system sometimes reports clouds in the FOV, even if there exists no cloud in the image. This is the reason why the PPV is 72.7% in the inside testing, while 69.9% in the outside testing. To be brief, our developed system does not miss any cloud event, but sometimes gives a false alarm.

Table 3 The True Positive Rate and Positive Predictive Value for Inside Testing

Predicted / Actual	Cloud	Non-cloud	Rate
Cloud	TP=253	FN=0	TPR=100%
Non-cloud	FP=95	TN=104	PPV=72.7%

Table 4 The True Positive Rate and Positive Predictive Value for Outside Testing

Predicted / Actual	Cloud	Non-cloud	Rate
Cloud	TP=179	FN=0	TPR=100%
Non-cloud	FP=77	TN=92	PPV=69.9%

5. CONCLUSIONS AND FUTURE WORK

In this paper, we develop an end-to-end multiscale deep learning approach to detecting clouds appearing in satellite images, which only use a CNN architecture in the whole process, and image pre-processing or post-processing are not required. Our developed system is able to detect clouds in different shapes and scales. The CNN architecture that we adopt is a kind of deep learning scheme. In this architecture, the skip-connection, concatenation, and layer dimension reduction are applied for better performance, particularly in promoting the system capability of detecting multiscale clouds.

However, the end-to-end approach using deep learning scheme requires GPU whose construction cost is relatively high. Although some embedded GPU architectures work with low watts, in the satellite environment, the lower power consumption is, the preferable system performance obtains. Therefore, in the future work, the FPGA implementation of the end-to-end deep learning approach will be considered for a lower power consumption and real-time computation solution.

ACKNOWLEDGEMENTS

The authors thank the National Space Organization of National Applied Research Laboratories of Taiwan (R. O. C.) for supporting this work in part under Grant NSPO-S-107100. And also thank Yu-Chi Huang, Yi-Chung Lan, Wei-Yu Chao, Hsiao-Chi Wu, Ting-Yu Lu, and Po-Yen Huang** who are the members of the joint research team affiliated to NTUST and NSPO** for enthusiastically participating the development of the real-time cloud detection system.

REFERENCES

[1] N. Pergola et al., "Improving volcanic ash cloud detection by a robust satellite technique," *Remote Sensing of Environment*, vol. 90, no. 1, pp. 1-22, 2004.

[2] D. Lu, "Detection and substitution of clouds/hazes and their cast shadows on IKONOS images," *International Journal of Remote Sensing*, vol. 28, no. 18, pp. 4027-4035, 2007.

[3] P. Li, L. Dong, H. Xiao, and M. Xu, "A cloud image detection method based on SVM vector machine," *Neurocomputing*, vol. 169, no. 2, pp. 34-42, 2015.

[4] T. Bai et al., "Cloud detection for high-resolution satellite imagery using machine learning and multi-feature fusion," *Remote Sensing*, vol. 8, no. 9, pp. 715-721, 2016.

[5] Deng et al., "Cloud detection in satellite images based on natural scene statistics and Gabor features," *IEEE Geoscience and Remote Sensing Letters*, vol. 16, no. 4, pp. 608-612, 2018.

[6] K. Alex, S. Il Ilya, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, no. 2, pp. 1097-1105, 2012.

[7] J. Redmon et al., "You Only Look Once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, pp. 779-788, 2016.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, pp. 770-778, 2016.

[9] W. Shang et al., "Understanding and improving convolutional neural networks via concatenated rectified linear units," in *Proceedings of the International Conference on Machine Learning*, New York, NY, pp. 2217-2225, 2016.

[10] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv:1804.02767 [cs.CV] 2018.