Deep Convolutional Neural Network for Cloud Coverage Estimation from Snapshot Camera Images

Article *in* Scientific online letters on the atmosphere: SOLA · January 2017

DOI: 10.2151/sola.2017-043

2 authors, including:

Ryo Onishi

Japan Agency for Marine-Earth Science Technology

68 PUBLICATIONS **329** CITATIONS

Deep Convolutional Neural Network for Cloud Coverage Estimation from Snapshot Camera Images

Ryo Onishi[1] and Daisuke Sugiyama[1]
[1] *Center for Earth Information Science and Technology, Japan Agency for Marine-Earth Science and Technology, Yokohama, Japan*

Corresponding author: Ryo Onishi, 3173-25 Showa-machi Kanazawa-ku, Yokohama Kanagawa 236-0001, Japan. E-mail: onishi.ryo@jamstec.go.jp.

Manuscript submitted on 7 September 2017 and accepted on 11 November 2017

Abstract

We have proposed a deep convolution neural network (CNN) approach for the accurate estimation of the cloud coverage (CC) from images captured by a consumer camera, i.e., snapshot pictures. This CNN can successfully estimate the CC to within the level of the inherent error in the training dataset. A segmentation-based method using a linear support vector machine (SVM) is shown to be unable to distinguish between water surfaces and the sky, while the present CNN can correctly distinguish between them, possibly because the CNN can understand the positioning of components in the images; the sky is over a water surface. The present CNN can also be applied to photo-realistic computer-graphic (CG) images from numerical simulations. Comparisons between the CNN estimates for camera images and for the CG images can provide useful information for data assimilation, and thus contribute to numerical weather forecasting. The CC is a sort of far-field (remote) information. The present CNN has the potential to allow consumer cameras to be used as remote weather sensors.

1. Introduction

To realize a "smart" society, it is key to utilize Internet of Things (IoT) sensor data for monitoring and forecasting. For example, IoT data can be used to improve numerical weather simulations, although there are several challenges that must first be overcome, e.g., the inhomogeneous quality of the IoT data. Another challenge is that most IoT sensors are located near ground surfaces and do not provide information about higher altitudes. For numerical weather simulations, observational data at high altitudes or vertical profiles are crucial, although the IoT sensors usually cannot provide such data. Remote sensors can be used, such as radars and lidars, but are very costly. In contrast, digital consumer cameras, which are often connected to the Internet, are cheap, ubiquitous, and moreover able to capture images of the sky and clouds, which constitute a potentially useful source of information about high altitudes. If we can extract information about the sky from such camera images, these cameras can be considered as cheap remote sensing devices and the data provided by them can be used for data assimilation for numerical weather prediction.

Currently, a major research topic concerning the extraction of sky information from camera images is how to accurately estimate cloud coverage (CC). There have been many attempts to extract CC from whole-sky images captured by fixed ground-based

cameras via segmentation-based (i.e., pixel-based) schemes (Dev et al. 2016; Manger and Pagel 2015; Chow et al. 2011; Li et al. 2011; Tzoumanikas et al. 2013; Gonzales et al. 2012; Ghonima et al. 2012; Yang et al. 2015), which classify individual pixels into sky, cloud, and non-sky (other than sky or cloud) segments based on the color of the pixel. It is easy to extract sky areas from fixed-frame images since the sky/non-sky boundaries can easily be determined by comparing multiple images. Thus, the main challenge for such schemes is how to distinguish cloud in the sky. The range of colors present in such fixed-frame images is limited, making CC estimation easier.

To the best of our knowledge, however, no attempt has been made to use moving-frame images for CC estimation although many of the consumer cameras are not fixed unlike fixed point cameras for security or for special purposes. This study therefore aims to estimate CC from images of skies captured by IoT consumer cameras. A deep learning approach using a convolutional neural network (CNN) is presented here (Vaillant et al. 1994). CNNs are biologically inspired deep learning models and have been attracting attention in the field of computer vision because of their high accuracy in image recognition tasks, such as the ImageNet Large Scale Visual Recognition Challenge (Krizhevsky et al. 2012; Simonyan and Zisserman 2014; Szegedy et al. 2015). The effectiveness of the present methodology is compared against a conventional segmentation-based scheme. The key finding of this study is that, unlike the segmentation-based scheme, the CNN approach can consider the inter-segmentation relation. The CC is a sort of far-field (remote) information. The present CNN has the potential to allow consumer cameras to be used as remote weather sensors.

2. Methods

Cloud coverage (CC)

The meteorological observatories of the Japan Meteorological Agency (JMA) record CC on an integer scale, from 0 to 10, at 1-hour intervals. The CC is judged from the cloud cover rate of the whole sky. In contrast, we here define the CC of the fraction of the sky captured in digital camera images. We are therefore considering the ‘partial’ CC. The area of sky outside of the image or obscured by objects is excluded from our CC estimation.

Although CC can be estimated in terms of the percentage of cloud cover, from 0% to 100%, here we adopt the 0–10 classification scale of the JMA. The 0–10 discrete representation is more practical for human labelers when labeling the teaching images for learning than the continuous representation.

Support Vector Machine (SVM)

We adopted a two-step approach for the classification of pixels into sky, cloud and non-sky segments: In the first step, the pixel is classified into sky-cloud or non-sky segments. In the second step, the sky-cloud pixel is classified into sky or cloud segments.

To determine the separation threshold between sky-cloud and non-sky segments, we collected 120 camera images that contain only sky and clouds, i.e., without including

non-sky objects such as animals, plants, buildings, and topological features. Then we analyzed them in the hue, saturate, and value (brightness) (HSV) space, which is a common cylindrical-coordinate representation of pixels in the RGB color space. We successfully obtained a typical range of values for the sky-cloud segment (see Supplementary Figure A1).

We distinguish cloud from sky with the aid of a linear support vector machine (SVM). We chose the following channels for the SVM training: red (r), green (g), blue (b), saturation in the cylinder coordinate (s_{cyl}), saturation in the cone coordinate (s_{cone}), value (brightness) (v), and b-r difference (br), which is defined as $br = (b - r)/(b + r)$. Note that r, g, and b are all normalized to values between 0 and 1. These channels were selected because they had significant variance among sky-cloud images. We collected digital images that contain only sky and only clouds, and used them as input for the SVM. The SVM provided the following criterion V_{sc}:

$$V_{sc} = -6.28r + 0.454g - 4.11b - 1.81s_{cyl} - 4.04s_{cone} + 8.88v + 1.53br + 0.586. \quad (1)$$

If $V_{sc} < 0$, then the pixel is judged to be a cloud pixel.

Convolutional Neural Network (CNN)

Figure 1 summarizes the architecture of the CNN considered here. The CNN outputs the CC estimate, ranging from -0.5 to 10.5 in the continuous representation, for the input image. TensorFlow (Abadi et al. 2016), an open source library provided by Google, is used to realize the present CNN architecture, which consists of 5 convolutional layers (*conv.*1, ..., *conv.*5), 5 pooling layers (*pool.*1, ..., *pool.*5), a fully connected layer (*fc*), and an output layer (*out*). The convolutional layer extracts features from images with learnable filters convolved across the width and height of the input image. An activation function is used along with each filter to produce an activation map of that filter. Pooling is a form of down sampling. Intuitively, the exact location of a feature is less important than its rough location relative to other features. The pooling layer serves to reduce the spatial size of the representation with mostly discarding the information about absolute positioning. The fully connected layer conducts high-level reasoning based on the output from the convolutional and pooling layers. The term 'fully' implies that the all the available elements are used for the reasoning. Each layer used in this study is specifically described in the following sections.

The input image has 128 × 128 pixels with 3 color channels (red, green, and blue); i.e., the input information has 128 × 128 × 3 dimensions. The number of dimensions is determined by the operation layers. A convolutional layer maps an $h \times h \times n$ image onto an $h \times h \times n'$ image by using a convolutional filter with dimensions $3 \times 3 \times n$. The output image is expressed as

$$u_{i,j,k} = f\left(b_k + \sum_{i'=-1}^{1} \sum_{j'=-1}^{1} \sum_{k'=1}^{n} z_{i+i',j+j',k'} \cdot w_{i',j',k',k} \right), \quad (2)$$

where $u_{i,j,k}$ and $z_{i,j,k}$ are the (i, j)-th pixel of the k-th channel in the output image and the input image for the layer, respectively, $w_{i',j',k',k}$ is the kernel for the k-th output channel, b_k is the bias for the k-th channel, and $f(x)$ is an activation function. Saturating functions, such as $f(x) = (1 + e^{-x})^{-1}$, are much slower than non-saturating functions in terms of

training time. We therefore employed the Rectified Linear Unit (ReLU), a non-saturating function, for the activation function:

$$f(x) = \max(x, 0). \quad (3)$$

Zero-padding on the input image is used to preserve the image width and height; namely, $z_{i,j,k}$ is supposed to be zero if $i \notin [1, h]$ or $j \notin [1, h]$.

The pooling layer used is a max pooling with a 2 × 2 filter and stride 2, which downsamples an $h \times h \times n$ image into an $h/2 \times h/2 \times n$ image as

$$u_{i,j,k} = \max\left\{ z_{i'+2i,j'+2j,k} \mid i', j' = \{0 \text{ or } 1\} \right\}. \quad (4)$$

In order to reduce overfitting in *fc*, we employed the dropout regularization method (Srivastava et al. 2014). The convolutional, pooling and fully-connected layers generate the set of 512 elements in the present CNN, while the final output we need is a single CC value. The output layer obtains the final output through the inner product of weight vector, whose weighting is tuned through the learning.

Training data preparation

The present CNN is a supervised machine learning approach, and therefore requires several labeled images for the training phase. We collected snapshot camera images from the Flickr website using the search word "sky" in the Flickr API, as well as our own digital camera albums. Note that the images presented in this paper were all selected from our own albums and not from other Flickr users, to avoid copyright infringement. The collected images were cropped to squares: The center part of the image was extracted by cropping if a camera image was wider horizontally than vertically, while the square was extracted from the upper part of the image if the vertical size was larger. The cropped images were resized to 128 × 128 pixels.

Figure 2 shows the selected training image samples together with their labels. It should be emphasized that the images are not pure-sky images and contain non-sky parts, such as houses and topographical features. The collected images were labeled by four human labelers. The labelers were not experts at CC labeling but had received a common orientation to try to standardize the labeling. When labeling, the labelers excluded images that were unsuitable for our purpose such as illustrations, unclear (out-of-focus) images, monochrome images, images with extremely small sky areas, images taken at night, retouched images, and images taken from airplanes. We constructed a dataset, $D = \{(\boldsymbol{x}_1, d_1), (\boldsymbol{x}_2, d_2), ..., (\boldsymbol{x}_N, d_N)\}$, where $\boldsymbol{x}_i$ denotes a camera image, and d_i is the CC (i.e., an integer from 0 to 10) of $\boldsymbol{x}_i$. We prepared 1,776 images with labels. The three-fourths of the prepared images were used for the training and the rest images for cross validations. The number of training images in each CC category was nonuniform and the CC = 0 (no clouds) category contained the largest number of images (see Fig. 5). We made an adjustment to equalize the number of images in each category by adopting the real-time data augmentation, which increases the number of training data through artificial image generations with a set of transformations (e.g., shifting, shearing, and zooming whose parameters are randomly chosen) applied on the input images.

The labels in D contain inherent errors due to human subjectivity. To evaluate the level of inherent error in the labels, 551 images out of the dataset D were reevaluated (relabeled) by the same human labelers. Let $D_{\text{relabel}} = [(\boldsymbol{y}_1, \{e_1^{(1)}, e_2^{(1)}, ...\}), (\boldsymbol{y}_2, \{e_1^{(2)}, e_2^{(2)}, ...\}), ..., (\boldsymbol{y}_{N'}, \{e_1^{(N')}, e_2^{(N')}, ...\})]$ be the dataset obtained by relabeling, where $\boldsymbol{y}_i$ is a relabeled image, $e_1^{(i)}$ is the original label of $\boldsymbol{y}_i$, and $e_j^{(i)}$ ($j \geq 2$) is a reevaluated label. The

size of dataset S_{relabel}, denoted as N', is 576. In this dataset, 25 images have three labels, while 551 images have two labels. The variance of the labels in D_{relabel} was 2.15, which was obtained by

$$\frac{1}{N'}\sum_{i=1}^{N'}\frac{1}{n^{(i)}-1}\sum_{j=1}^{n^{(i)}}\left(e_j^{(i)}-\frac{1}{n^{(i)}}\sum_{k=1}^{n^{(i)}}e_k^{(i)}\right)^2, \tag{5}$$

where $n^{(i)}$ is the number of labels in the i-th item of D_{relabel}. This value is an estimate of the inherent (inevitable) error in the labels in D due to human subjectivity.

3. Results and discussion

Figure 3 shows the learning curve of the CNN. It took 10.4 hours to complete the training phase, with up to 300,000 training cycles, on a single Tesla K40 GPU board. The mean square error (MSE) decreased to just above 3 by the end of the training phase. The minimum MSE is 3.0 at 268,000 training cycles and the MSE often comes close to 3.0, but does not go below. This means that the CNN has become saturated and cannot be improved beyond this level with the training data. It should be noted that the attained error level (nearly 3) is already close to the inherent error level in the teaching labels (i.e., 2.15, as discussed in the previous subsection).

Figure 4 shows the confusion matrices of the SVM prediction and the CNN prediction. Each row of the confusion matrix represents the instances in a predicted class while each column represents the instances in an actual class, and high values in the diagonal elements indicate accurate estimates in the confusion matrix. The confusion matrix of the CNN prediction shows much higher values in the diagonal elements than that of the SVM prediction, showing that the CNN has a higher prediction skill than the SVM. In particular, we see clear differences in clear sky (CC=0) and overcast (CC=10) cases: The CNN shows very good skills for the two opposed cases, while the SVM shows poor skills. The square error of the SVM predictions was 10.8, while that of the CNN predictions was 3.0. The error level of the CNN predictions was as small as the inherent (inevitable) errors in the training data due to human (labeler) subjectivity (i.e., 2.15). We made several trial-and-error attempts to improve the algorithm for the SVM system, but we could not obtain a better algorithm. A major source of error in the SVM predictions came from distinguishing between sky and water surfaces (ocean, lakes, and rivers) and is discussed later.

Figure 5 shows the cloud coverage (CC) averaged over 7 years from 2010 to 2016, at the meteorological observatories in seven major Japanese cities; Sapporo, Sendai, Tokyo, Yokohama, Nagoya, Osaka, and Hakata. On average, the sky is fully covered by clouds (i.e., CC = 10) in around 30% of images, and clear (i.e., CC = 0 or 1) in about 10% of images. Interestingly, clear sky images accounted for 25% of images in the training data (snapshot camera images). The snapshot camera images were not limited to Japanese cities, and therefore we cannot draw any definite conclusions, but the relatively high chance of clear sky in the snapshot images may be due to a tendency for people to take pictures more frequently under clear skies. It should be noted that, after the standardization procedure described above, the adjusted training dataset had a uniform probability for each CC category. When rescaled for the CC probabilities averaged over the seven Japanese cities, the MSE of the SVM and the CNN became

12.6 and 2.49, respectively. That is, the CNN shows a higher prediction accuracy for either situation.

We surveyed the cause of the poor prediction skill of the SVM system. We confirmed that the major source of error is the misclassification of water surfaces as sky due to their similarity in color. Figure 6 shows an example of the SVM failing in the CC estimation; the estimated CC was 5.9, while the labeled CC was 4. The right-hand figure shows that the major fraction of sea was classified as sky (blue) by the SVM. In contrast, the CNN successfully estimated the CC as 4.3. To investigate how the CNN correctly distinguished between sea and sky, we rotated the image by 180 degrees (i.e., turned the image upside-down) as in the lower panel. The SVM estimate was unchanged, but the CNN estimation accuracy became significantly lower (6.1). Here, we show only one example, but this behavior was widely observed in the images with water surfaces. We therefore infer that the CNN recognizes that the sky is in the upper region of an image when water surfaces are also present.

The results confirm that the present CNN can estimate the CC from consumer camera images. To use the CC information for data assimilation for numerical weather forecasting, we have to obtain the CC information from numerical simulations. Kawahara et al. (2015) developed a simple but powerful visualization algorithm that can provide photo-realistic (i.e., computer graphic, CG) images from three-dimensional simulation results. The downward shortwave radiation is utilized to represent the shade on clouds, to make the visualized cloud look realistic. Figure 7 shows an actual snapshot image with CC around 8 and a CG image with CC around 5 from a high-resolution numerical weather simulation. The CC estimates by the CNN were 8.1 and 4.5, respectively. This means that the present CNN can provide accurate CC estimation from CG images as well as from actual camera images. The comparison of the CC estimates for the camera and CG images can provide a measure of the reliability of the simulation. This reliability measure can then be used for data assimilation.

4. Conclusions

We have proposed a deep convolution neural network (CNN) system, i.e., a so-called deep learning system, for robust estimation of cloud coverage (CC) for skies captured in consumer camera images. The frames of such camera images are not fixed and contain non-sky objects, and their color ranges change from frame to frame, making CC estimation difficult. The present CNN, however, can successfully estimate CC to within an error level comparable to the error inherent in the training dataset. We compared the performance of our proposed method against that of a support vector machine (SVM), which is a widely used machine learning approach for image segmentation. The SVM system does not work well for the present image-processing task since it often misclassifies water surfaces as sky. In contrast, the CNN does not suffer from this problem, probably because the CNN can understand that in images containing water surfaces, the sky will be in the upper part of the image. For fixed-point camera images, for which the boundary between the sky-cloud and non-sky segments is fixed and easily determined, the SVM would work well. One strength of the present CNN is that its application is not limited to fixed-point camera images.

The CNN can be applied to CG images from numerical simulations. Then, comparison of CC estimates from consumer camera images and from CG images can be

used for data assimilation. This means that consumer cameras can be used as remote sensors in the sense that the CC is far-field (sky) information. Usually remote weather sensors, such as radars and lidars, are very expensive, but the use of the present CNN could enable cheap remote-sensing observation. A possible area for future work is whether IoT sensor data used with the present CNN could be used in local high-resolution weather simulations.

Acknowledgements

The authors are grateful to Dr. Akio Kawano for his advice on machine learning techniques.

Supplement

Supplement 1 shows the color distribution of the pixels in sky-cloud images in the HSV space, on which the pixels in camera images are classified into sky-cloud and non-sky segments.

References

Abadi, M., P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. A. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zhang, 2016: TensorFlow: A system for large-scale machine learning, *Proc. of 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, **16**, 265-283.

Chow, C. W., U. Bryan, L. Matthew, D. Anthony, K. Jan, S. Janet, and W. Byron, 2011: Intra-hour forecasting with a total sky imager at the UC San Diego solar energy testbed, *Solar Energy*, **85**, 2881--2893.

Dev, S. and B. Wen, Y. H. Lee, and S. Winkler, 2016: Machine Learning Techniques and Applications For Ground-based Image Analysis, arXiv:1606.02811.

Flickr, https://www.flickr.com/

Ghonima, M. S., B. Urquhart, C. W. Chow, J. E. Shields, A. Cazorla, and J. Kleissl, 2012: A method for cloud detection and opacity classification based on ground based sky imagery, *Atmospheric Measurement Techniques*, **5**, 2881-2892.

Gonzales, Y., C. L'opez, and E. Cuevas, 2012: Automatic observation of cloudiness: analysis of all-sky images, *Proc. of WMO Technical Conference on Meteorological and Environmental Instruments and Methods of Observation,* **3**.

Kawahara, S., R. Onishi, K. Goto, and K. Takahashi, 2015: Realistic Representation of Clouds in Google Earth, *Proc. of Symposium on Visualization in High Performance Computing in SIGGRAPH ASIA*, doi:10.1145/2818517.2818541.

Krizhevsky, A., I. Sutskever, and G. E. Hinton, 2012: Imagenet classification with deep convolutional neural networks, *Advances in neural information processing systems*, 1097-1105.

Li, Q., W. Lu, and J. Yang, 2011: A hybrid thresholding algorithm for cloud detection on ground-based color images, *Journal of atmospheric and oceanic technology*, **28**,

1286-1296.

Manger, D., and F. Pagel, 2015: Camera-based forecasting of insolation for solar systems, *SPIE/IS&T Electronic Imaging*, International Society for Optics and Photonics, 94050M (6 pages).

Simonyan, K. and A. Zisserman, 2014: Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556.

Srivastava, N., G. Hinton, E. Geoffrey, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, 2014: Dropout: a simple way to prevent neural networks from overfitting, *Journal of Machine Learning Research*, **15**, 1929-1958.

Szegedy, C., W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, 2015: Going Deeper With Convolutions, *Proc. of The IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 1-9.

Tzoumanikas, P, A. Kazantzidis, A. F. Bais, S. Fotopoulos, and G. Economou, 2013: Cloud detection and classification with the use of whole-sky ground-based images, *Advances in Meteorology, Climatology and Atmospheric Physics*, Springer, 349-354.

Vaillant, R., C. Monrocq, and Y. L. Cun, 1994: Original approach for the localisation of objects in images, *IEE Proceedings - Vision, Image and Signal Processing*, **141**, 245-250.

Yang, J., Q. Min, W. Lu, W. Yao, Y. Ma, J. Du, T. Lu, and G. Liu , 2015: An automated cloud detection method based on the green channel of total-sky visible images, *Atmospheric Measurement Techniques*, **8**, 4671-4679.

Figures

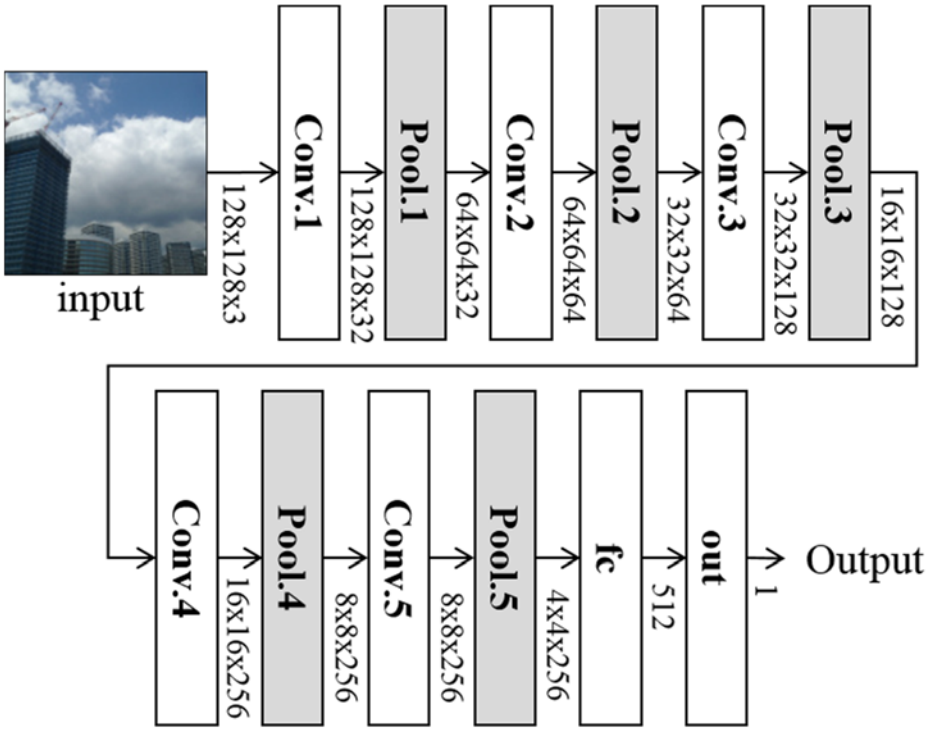


Fig. 1. Illustration of the architecture of the present CNN, which explicitly shows the sizes of the matrices; width × height × channels.


Label: 0


Label: 5


Label: 6


Label: 10

Fig. 2. Examples of snapshot camera images and their labels. The label ranges from 0 (no clouds, clear sky) to 10 (fully covered by clouds, overcast).

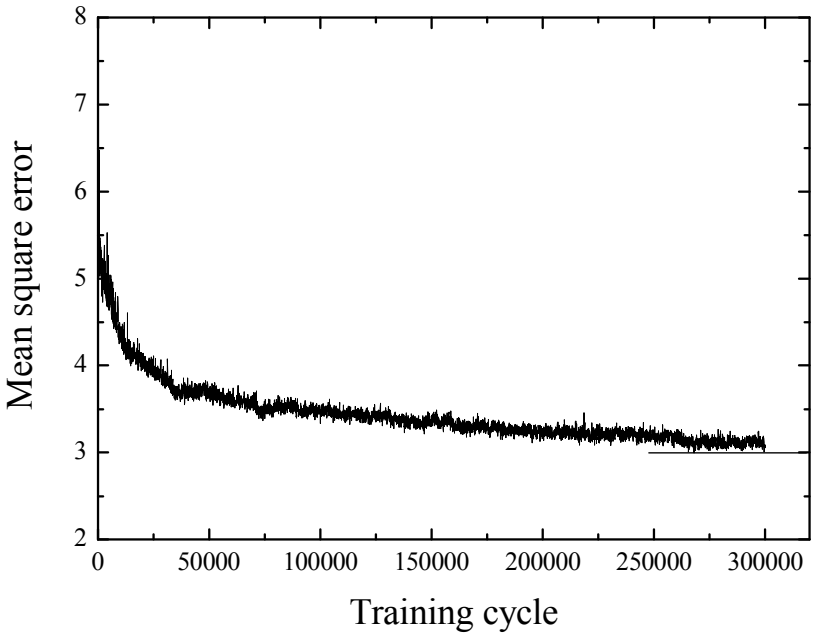
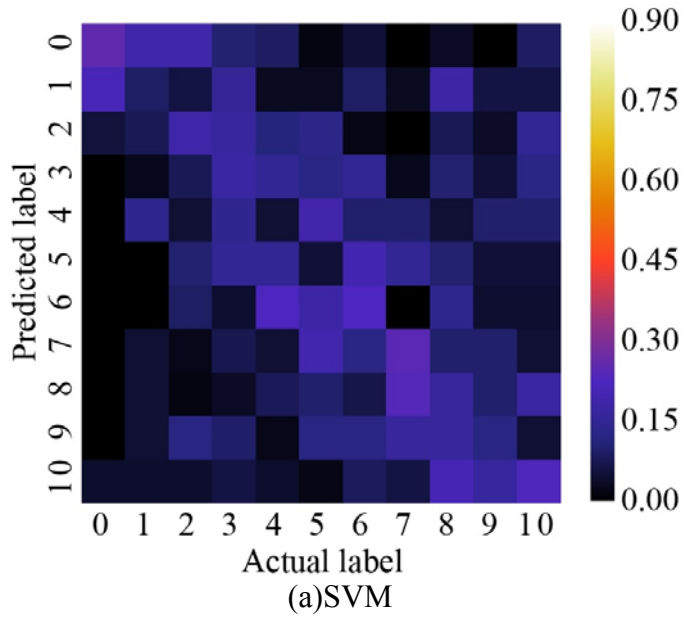
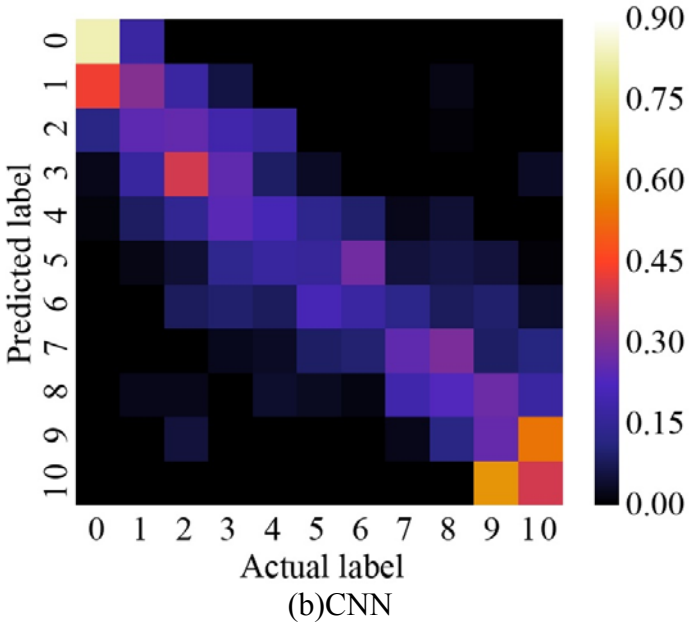


Fig. 3. Learning curve of the present CNN. The mean square error saturates at 3.0 after 268,000 training cycles.



(a)SVM



(b)CNN

Fig. 4. Confusion matrix of the (a) SVM and (b) CNN. The mean square error is (a) 12.7 and (b)3.0.

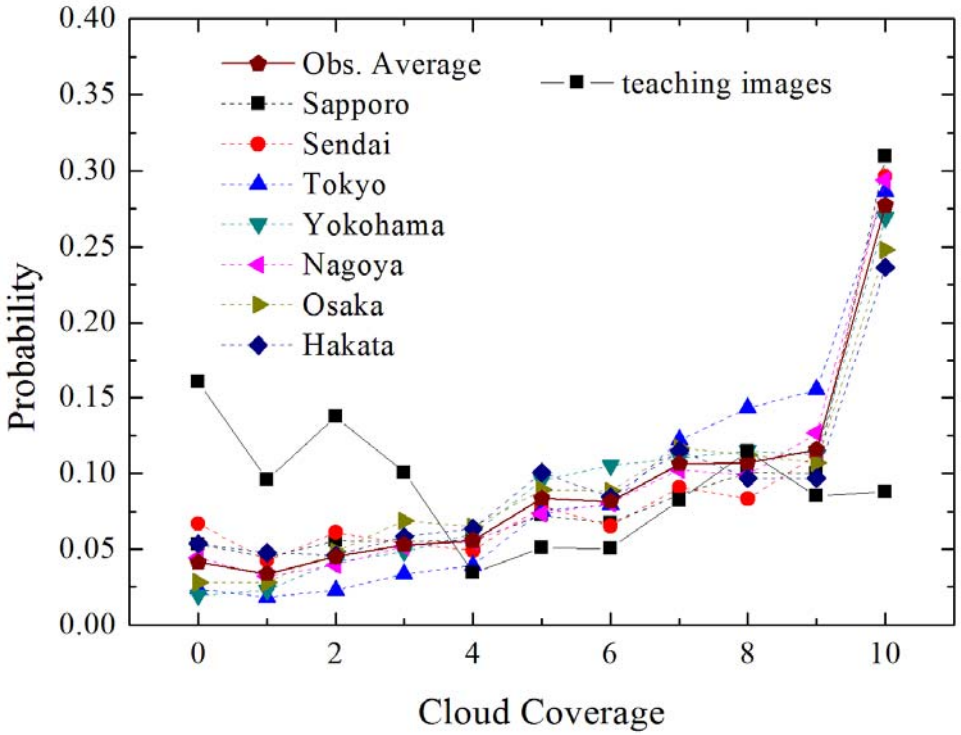
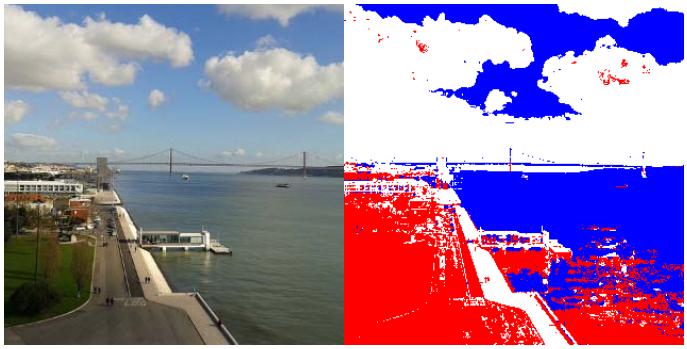
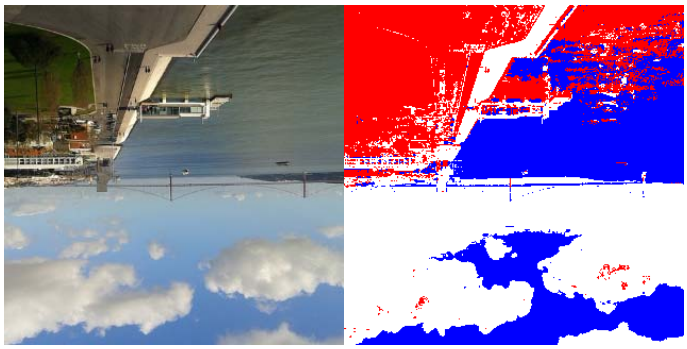


Fig. 5. Cloud coverages at the meteorological observatories in major Japanese cities averaged over 7 years from 2010 to 2016, together with the cloud coverage in the training images.



(a) original



(b) upsidedown

Fig. 6. Typical snapshot with cloudy sky over the sea. The upper panels are the correctly positioned images (i.e., original images), while the lower panels are the upside-down images. The right images show sky (blue), clouds (white) and non-sky (red) segmented by the present SVM. The label was 4, while the SVM estimate was 5.9 for both images. The CNN estimate was 4.3 for the correctly positioned image and 6.1 for the upside-down image.

(a) (b)

Fig. 7. CNN estimates of the CC for (a) an actual snapshot image and (b) a CG image from a numerical weather simulation. The CC estimates by the CNN were (a) 8.1 and (b) 4.5, respectively.

Supplementary figures

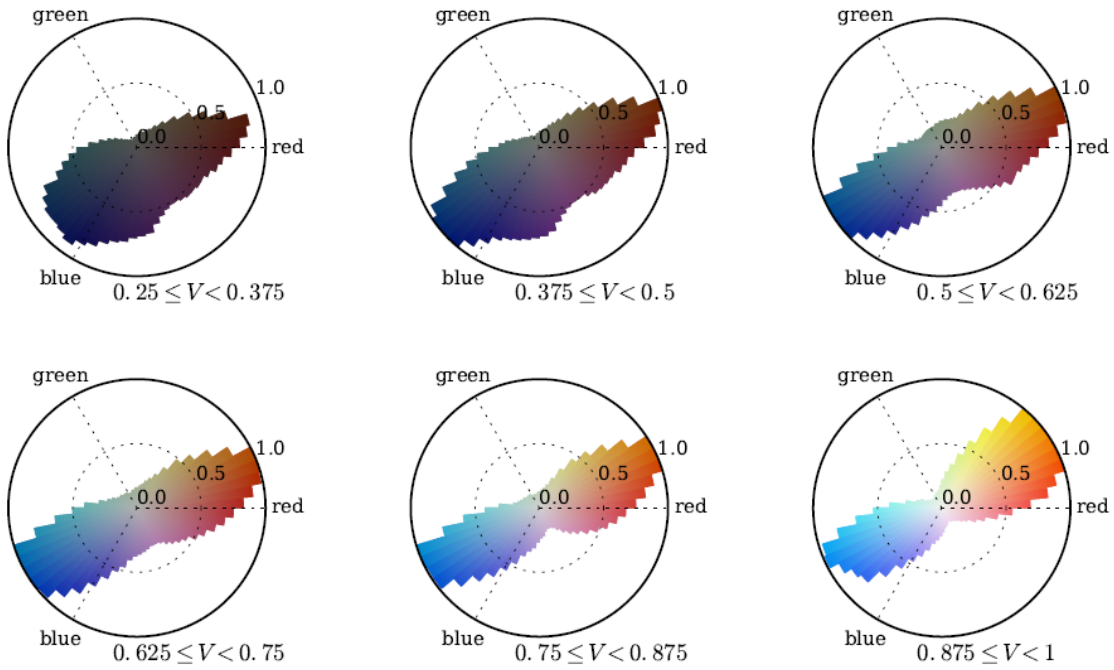


Fig. A1: Color distribution of the pixels in sky-cloud images in the HSV space. Each panel shows the slice at a given saturation level; 0.2 < V < 0.375, 0.375 < V < 0.5, 0.5 < V < 0.625, 0.625 < V < 0.75, 0.75 < V < 0.875, and 0.875 < V < 1. An image pixel is classified as sky-cloud segment if its color is within the marked area.