# EFFICIENT CLOUD DETECTION IN REMOTE SENSING IMAGES USING EDGE-AWARE SEGMENTATION NETWORK AND EASY-TO-HARD TRAINING STRATEGY

*Kun Yuan*[1,2], *Gaofeng Meng*[1], *Dongcai Cheng*[1], *Jun Bai*[3], *Shiming Xiang*[1] *and Chunhong Pan*[1]

[1] National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences
[2] School of Computer and Control Engineering, University of Chinese Academy of Sciences
[3] Research Center for Brain-inspired Intelligence, Institute of Automation, Chinese Academy of Sciences

## ABSTRACT

Detecting cloud regions in remote sensing image (RSI) is very challenging yet of great importance to meteorological forecasting and other RSI-related applications. Technically, this task is typically implemented as a pixel-level segmentation. However, traditional methods based on handcrafted or low-level cloud features often fail to achieve satisfactory performances from images with bright non-cloud and/or semitransparent cloud regions. What is more, the performances could be further degraded due to the ambiguous boundaries caused by complicated textures and non-uniform distribution of intensities. In this paper, we propose a multi-task based deep neural network for cloud detection in RSIs. Architecturally, our network is designed to combine the two tasks of *cloud segmentation* and *cloud edge detection* together to encourage a better detection near cloud boundaries, resulting in an end-to-end approach for accurate cloud detection. Accordingly, an efficient sample selection strategy is proposed to train our network in an easy-to-hard manner, in which the number of the selected samples is governed by a weight that is annealed until the entire training samples have been considered. Both visual and quantitative comparisons are conducted on RSIs collected from Google Earth. The experimental results indicate that our method can yield superior performance over the state-of-the-art methods.

***Index Terms***— Cloud detection, Segmentation network, End-to-end, Edge-aware, Easy-to-hard

## 1. INTRODUCTION

With the rapid development of remote sensing photograph acquisition technology, obtaining high resolution remote sensing images currently is not a difficult task. However, as clouds cover more than 50% surface of the earth [1], many meaningful ground objects are often occluded or contaminated by the clouds in a captured remote sensing image, which leads to great loss of availability of the image. Accordingly, cloud detection is often the first step in the processing flow of optical remote sensing images. Besides, by providing the estimation of cloud distributions, cloud detection can also find many applications in weather forecast, urban vehicle condition rendering and house detection.

Cloud detection is a very challenging problem. To detect cloud regions, one needs to first recognize the cloud layers, and then segment the remote sensing images into two classes (cloud regions and non-cloud regions). However, for complex images that have bright non-cloud regions (ice surface, desert and breakers along ashore, etc.) and semitransparent cloud pixels, current techniques usually fail to provide satisfactory performances. Besides the detection accuracy, high efficiency and automaticity (no user interactions) should be taken into consideration when processing high resolution images.

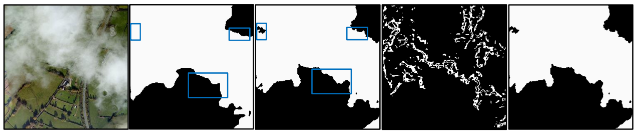

**Fig. 1**: Results of different detection methods. (a) Cloud image. Results of (b) SegNet [13], and (c) our method. Especially, (d) is the edge detection result by our method. (e) Ground truth. Our method can obtain few mislabels and more accurate edges, such as the comparison of the rectangle regions in (b) and (c).

In the literature, a number of cloud detection methods have been proposed. Most of these methods [2, 3, 4] are developed for moderate spatial resolution sensors such as high resolution radiometer and moderate resolution imaging spectroradiometer. More in detail, some traditional and popular techniques, which require a preliminary careful study of the spectral properties of clouds, rely on multiple threshold selection [5, 6, 7]. Other widely used supervised algorithms are based on support vector machines(SVMs), which have been successfully applied to hyperspectral image classification in general [8, 9] and to cloud detection in particular [10, 11]. They analysed luminance feature and texture features. Typically, An et al.[12] trained cloud detectors to acquire saliency maps, and refined them using optimal threshold algorithm. It should be noted that some of these traditional methods are time consuming. Meanwhile, distinguishing clouds from ice surface is still challenging for them.

With the wide application of deep learning, architectures based on convolutional neural networks are also applied to tackle cloud detection. Shi et al. [14] proposed to use SLIC [15] to cluster the image into small superpixels and trained CNNs for a binary class classification task. Meanwhile, deep segmentation networks such as FCNs [16], SegNet [13], DeconvNet [17] can also been used to cloud detection task. However, some existing problems need to be tackled when using deep learning methods. First, acquisition of labeled cloud images consumes large amounts of time and efforts. Second, due to the fuzzy boundary of cloud, traditional methods may have difficulties in distinguishing between semitransparent cloud pixels and background around boundaries. Third, how to achieve faster convergence when training image samples with various background conditions is still a challenging task.

This work was supported by the National Natural Science Foundation of China (Grants No. 91338202, 61370039, 91646207 and 61403377), the Beijing Natural Science Foundation (Grant No. 4162064), and the National 863 projects (Grant No. 2015AA042307).

In this paper, we propose a multi-task based deep neural network for cloud detection in remote sensing images. This method is distinguished by the following two main contributions:

- An edge-aware network is proposed which combines cloud segmentation and cloud edge detection together to encourage better detection results near cloud boundaries, resulting in an end-to-end approach for accurate cloud detection.
- A training strategy based on easy-to-hard sample selection is proposed for efficient training of the network. The number of selected samples is governed by a weight that is annealed until the entire training samples have been considered. Experimental results show that this strategy can achieve better convergence and improve segmentation results.

## 2. EDGE-AWARE SEGMENTATION NETWORK AND ITS TRAINING STRATEGY

Different from other datasets, our dataset contains edge information which is beneficial for detection improvement. To make full use of data information, an edge-aware network and easy-to-hard training strategy are proposed.

### 2.1. Edge-aware Network for Cloud Detection

Researches on the visualization of deep convolutional neural networks show that former layers of networks generally contain features much related to image edge information. Based on this observation, an edge-aware network is developed for cloud detection to obtain better results around cloud boundary regions.

In our network, cloud edge detection and cloud segmentation are combined. By introducing edge detection subtask, edge information is used to regularize the former convolutional layers as well as enhancing segmentation results near boundary regions. They can also be considered as the increase of the weights of true edge points in the segmentation network. As illustrated in Fig. 2, our network mainly consists of edge detection and segmentation parts. Feature maps extracted from the first two convolutional layers are merged, then convolutional and softmax layer are adapted to obtain edge information. Deconvolution layer is added to the second convolutional layer to keep image size and filter numbers the same as the first convolution layer, which guarantees the fusion process smoothly. In the segmentation part, symmetrical convolutional and pooling layers make up an encoder-decoder structure. Feature maps from the last convolutional layer are fused with edge information. The final convolutional output is fed to a multi-class softmax classifier to produce class probabilities for each pixel independently.

We implement totally 4 convolutional layers in encoder part and the same in decoder part. Each convolutional layer in the encoder network performs convolution with a filter bank to produce a set of feature maps. Batch normalization (BN) [18] is applied. Then an element-wise rectified-linear (ReLU) $max(0, x)$ [19] is applied. Following that, max-pooling with $2 \times 2$ window and stride of 2 is performed, and the resulting output is sub-sampled. Max-pooling is used to achieve translation invariance over small spatial shifts in the input image. After the last 64-channel convolutional layer, we put another 2-channel convolution layer as well as the softmax layer to obtain final prediction maps for our two-class problem.The output of the softmax classifier is a 2-channel maps of probabilities where 2 is the number of classes (cloud and non-cloud). The predicted label corresponds to the class with maximum probability at each pixel. Experiments demonstrate that these implementation preferences are efficient and effectual.

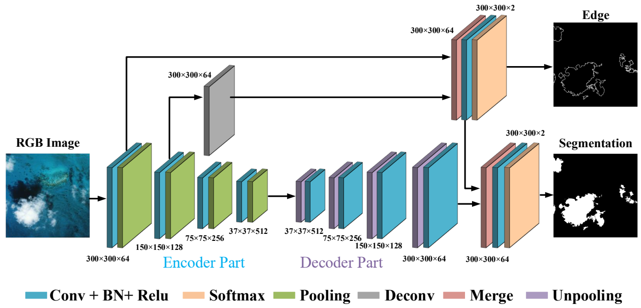

**Fig. 2**: Illustration of the overall architecture of the proposed model. Feature maps extracted from the first two convolution layers in encoder component are merged to get the edge detection result. Meanwhile, edge information is used to enhance the segmentation performance by merging with the last convolution layer. Input size for each layer is fixed, for instance, 300 × 300 × 64 means that input of the layer has 64 channels with pixels of 300 × 300.

### 2.2. Easy-to-hard Training Strategy

Due to the uneven distribution of cloud, in which types of the backgrounds vary from sample to sample, the difficulty for learning samples is different. Inspired by a method called *Self-Paced Learning*(SPL) proposed by Kumar et al. [20], we start with learning easier samples of cloud data, and then gradually take more complex samples into consideration. This simple idea demonstrated to be beneficial in avoiding bad local minima and in achieving a better generalization result.

Given cloud dataset $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1, \mathbf{z}_1), \cdots, (\mathbf{x}_k, \mathbf{y}_k, \mathbf{z}_k), \cdots (\mathbf{x}_n, \mathbf{y}_n, \mathbf{z}_n)\}$, where $\mathbf{x}_k \in \mathbb{R}^{h \times w \times c}$ denotes the $k^{th}$ cloud image, and $\mathbf{y}_k, \mathbf{z}_k \in \mathbb{R}^{h \times w}$ represent its segmentation and edge label, $h, w, c$ depict height, width and channels of an image. The total number of samples is $n$. Let $f$ and $\mathbf{w}$ represent the learned function and model parameters for our multi-task network. Through network can we get edge and segmentation predictions denoted by $f_{edge}(\mathbf{x}_k, \mathbf{w})$ and $f_{seg}(\mathbf{x}_k, \mathbf{w})$. Respectively, for $k^{th}$ image sample, the segmentation loss is calculated by binary cross-entropy:

$$L_{seg} = -\frac{1}{N} \sum\nolimits_{i=1}^{N} \{ y_i \ln f_{seg_i}(\mathbf{x}_k, \mathbf{w}) + \\ (1 - y_i) \ln(1 - f_{seg_i}(\mathbf{x}_k, \mathbf{w})) \}, \quad (1)$$

where $N$ denotes total pixels in an image, $y_i$ represents the label value and $f_{seg_i}$ stands for the segmentation value of the point. $L_{edge}$ shares the *same format*. Then weight coefficients $\alpha, \beta$ are set for $L_{seg}$ and $L_{edge}$. So the final loss can be written as:

$$L(\mathbf{x}_k, \mathbf{y}_k, \mathbf{z}_k, f, \mathbf{w}; \alpha, \beta) = \alpha L_{seg} + \beta L_{edge}. \quad (2)$$

In our network, the goal is to jointly learn the model parameter $\mathbf{w}$ and the latent weight variable $\mathbf{v} = [v_1, \cdots, v_n]$ by minimizing:

$$\min_{\mathbf{w},\mathbf{v}} \sum\nolimits_{k=1}^{n} v_k L(\mathbf{x}_k, \mathbf{y}_k, \mathbf{z}_k, f, \mathbf{w}; \alpha, \beta) - \lambda \sum\nolimits_{k=1}^{n} v_k, \\ \text{s.t. } \mathbf{v} \in \{0, 1\}^n, \quad (3)$$

where $\lambda$ is a parameter for controlling the numbers of training samples. Eq. (3) indicates the loss of a sample is discounted by a weight. The objective of our method is to minimize the weighted training loss together with the negative $l_1$-norm regularizer $-||\mathbf{v}||_1 = -\sum_{k=1}^{n} v_k$ (since $v_k \geq 0$).

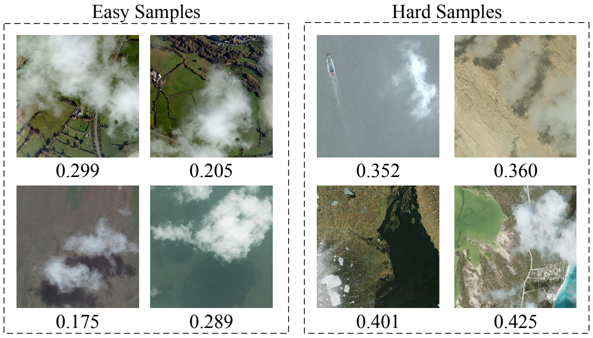

**Fig. 3**: An example of selected samples by our training strategy (the loss of each image is shown blew). On the left block, images with small loss are selected as "easy" samples. On the right block, images with (wave, shallow desert, ice and seabeach in order) are divided into "hard" samples.

Alternative Convex Search (ACS) [21] is used to solve Eq. (3). It is an iterative method for biconvex optimization, in which the cloud dataset is divided into two disjoint blocks depending on $\mathbf{v}$. In each iteration, samples in the selected block are employed to train the network and obtain the optimal $\mathbf{w}^*$, while the other block is fixed. With the fixed $\mathbf{w}$, the global optimum $\mathbf{v}^* = [v_1^*, \cdots, v_n^*]$ can be easily calculated by:

$$v_k^* = \begin{cases} 1, & L_{seg} < \lambda \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

There exists an intuitive explanation behind this sample-selected strategy:1)when updating $\mathbf{v}$ with a fixed $\mathbf{w}$, a cloud sample whose loss is smaller than a certain threshold $\lambda$ is taken as an "easy" sample, and will be selected in training ($v_i^* = 1$), or otherwise unselected ($v_i^* = 0$); 2) when updating $\mathbf{w}$ with a fixed $\mathbf{v}$, the network is trained only on the selected "easy" samples. Some easy and hard samples of cloud images are illustrated in Fig. 3. As can be seen that in easy samples, the cloud and background have apparent difference in color and texture. Conversely, for hard samples, due to the existence of noises such as ship wakes, ice surface and shallow color beach, the identification of cloud is difficult and the loss is high. The parameter $\lambda$ controls the number of samples fed into the network. When $\lambda$ is small, only "easy" samples with small losses will be considered. As $\lambda$ grows, more samples with larger losses will be gradually appended to train a more "mature" model.

In the training stage, we set $\alpha = 0.33, \beta = 0.67$ in Eq. 2 to enhance boundary pixel weight. Meanwhile, we initialize $\lambda = 0.35$ in Eq. 4 such that half the images are considered easy in the first training epoch, and let $\lambda$ grows such that $1.3$ times training samples are selected after each epoch.

## 3. EXPERIMENTS

In this paper, Keras [22] (a high-level neural networks library and capable of running on top of either TensorFlow or Theano) is used to run networks.

### 3.1. Data Description

The RSIs used in this paper are natural coloured and collected from Google Earth with a spatial resolution of $3.0 - 5.0$ m. There are 118 training images, 6 validation images, and 30 testing images, each with size around $688 \times 488$. We label the ground truths of all the images by hands. The ground truth of the edge is automatically calculated using 8-direction edge detection filters. Background scenes in the cloud images are diverse, including desert, sea, forest, island, city, ice land, etc.

To augment the training and validation sets, we firstly crop four corners and center of the original images with pixels of $300 \times 300$. Then the original images are random cropped to generate another 20 samples. We further rotate and mirror each of the above 25 samples to extend one sample to eight. In this way, the training and validation sets are augmented 200 times. There are totally 29119 training samples and 3900 validation samples, each with size $300 \times 300$. Before training, histogram equalization is adapted to samples instead of removing the mean value.

### 3.2. Comparisons and Evaluations

Five approaches are compared with our method: k-means [23], mean-shift [24], Progressive Refinement Scheme (PRS) [1], fully convolutional networks (FCNs) [16] and SegNet [13].

Evaluation indexes are defined to compare and analyze results. The right rate ($RR$), error rate ($ER$), false alarm rate ($FAR$), ratio of $RR$ to $ER$ ($RER$) and intersection over union ($IOU$) are used to evaluate the cloud detection results. They can be computed as:

$$\begin{aligned} &RR = \frac{TP}{TP + FN}, \quad IOU = \frac{TP}{TP + FP + FN}, \\ &ER = \frac{FP + TN}{TP + FP + TN + FN}, \quad RER = \frac{RR}{ER}, \\ &FAR = \frac{FP}{TP + FP + TN + FN}, \end{aligned} \tag{5}$$

where $TP$ represents the number of correctly detected cloud pixels. $FP$ denotes the number of non-cloud pixels mislabeled as cloud. $FN$ depicts the number of cloud pixels mislabeled as non-cloud. $TN$ is the number of correctly detected non-cloud pixels.

We compare the segmentation performances of six methods and some representative results are shown in Fig. 4. Traditional methods (K-means, Mean-shift and PRS are more prone to misclassify pixels with similar texture and intensity, such as rooftops in city, white sand beach. Specially, compared with traditional low-level features detection, our method can distinguish between ice land and cloud regions and results are shown in Fig. 5. FCNs and SegNet yield better results, but often have mislabels in cloud boundaries. Due to a high 8-times upsampling, FCNs generally causes huge information loss. As a result, their predicted segments tend to be blobby and lack fine object boundary details. In comparison, our method can obtain more spatially consistent results. Meanwhile, we achieve better results around boundaries than FCNs and SegNet due to the enhance of boundary responds.

We also calculate average RR, ER, RER, FAR and IOU on testing images and present them in Table 1. As can been seen from the table, our method achieves the best results on all of the five indices.

| | RR | ER | FAR | RER | IOU |
|---|---|---|---|---|---|
| K-means | 0.7644 | 0.1863 | 0.1196 | 9.1004 | 0.5454 |
| Mean-shift | 0.6475 | 0.1803 | 0.0981 | 7.2370 | 0.5018 |
| PRS | 0.8020 | 0.1222 | 0.0772 | 14.5679 | 0.6201 |
| FCNs | 0.8676 | 0.1323 | 0.0661 | 10.2407 | 0.7755 |
| SegNet | 0.9191 | 0.0809 | 0.0405 | 20.7698 | 0.8562 |
| Our method | **0.9334** | **0.0666** | **0.0333** | **24.9407** | **0.8786** |

**Table 1**: Segmentation results on five evaluation indexes.

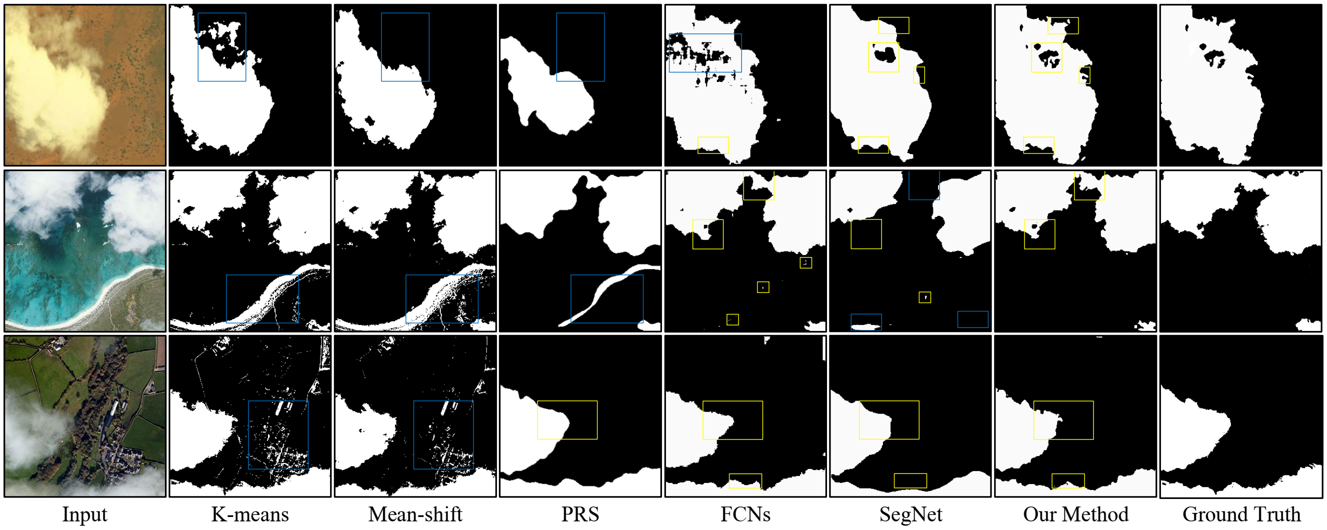

**Fig. 4**: Visual comparison of cloud detection results with six methods. In the blue rectangle regions, mislabeled pixels can be seen in semitransparent cloud, sandbeach, streets and houses regions when traditional methods are used. In the yellow rectangle regions, detection accuracy around edge regions has been distinctly improved. Compared with FCNs, spatial consistent can be maintained by our method.

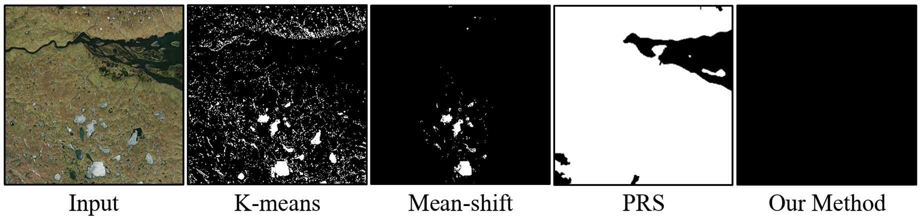

**Fig. 5**: Visual comparison of cloud detection for ice-covered regions. Due to the very similar brightness of ices and cloud regions, Kmeans, Mean-shift and PRS fail to distinguish them correctly. In comparison, our method can successfully distinguish these regions.

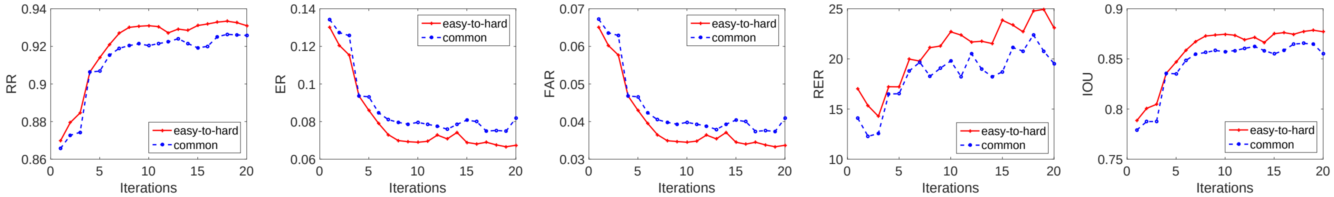

**Fig. 6**: Comparisons on Five Evaluation Indexes. Red lines denotes evaluations by our training strategy, while blue one represents non-strategy training method. Distinctly, better results can be achieved on all evaluation indexes by easy-to-hard strategy.

The running time of different methods are also calculated. For a 300 × 300 image, the times of k-means, mean-shift and PRS on 3.4 GHz Intel, Matlab R2015b are 3.40s, 0.85s and 1.45s respectively. On NVIDIA GTX 1080 GPU with 8-GB memory, FCNs, SegNet and our method need only 48ms.

### 3.3. Ablation Analysis

In order to verify the effectiveness of our sample selection method, we conduct comparative experiments on our network without easy-to-hard training strategy. We fed the same training set to both networks (with and without sample selection strategy), and check their performance on the validation set. The training curves can be seen in Fig. 6. Results show that by using easy-to-hard training strategy, our network can achieve better convergence and improve segmentation results. With common training strategy, we get segmentation results $RR = 0.9270, ER = 0.0730, FAR = 0.0365, RER = 19.7834, IOU = 0.8674$, which still work better than SegNet. The results demonstrate both edge-aware network and easy-to-hard training strategy work well in precision improvement.

## 4. CONCLUSION

In this paper, an edge-aware deep neural network has been proposed for cloud detection in RSIs. Compared with handcrafted or low-level features based cloud detection methods, our network combines cloud segmentation with cloud edge detection to encourage a better detection result near cloud boundaries. An easy-to-hard training strategy based on sample selection is also proposed to speed up the convergence of the network and improve the final segmentation results. Both visual and quantitative comparisons show that our method can yield superior results over the state-of-the-art methods.

## 5. REFERENCES

[1] Qing Zhang and Chunxia Xiao, “Cloud detection of rgb color aerial photographs by progressive refinement scheme,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 11, pp. 7264–7275, 2014.

[2] S. A. Ackerman, R. E. Holz, R. Frey, E. W. Eloranta, B. C. Maddux, and M. Mcgill, “Cloud detection with modis. part ii: Validation,” *Journal of Atmospheric & Oceanic Technology*, vol. 25, no. 7, pp. 1073–1086, 2008.

[3] Richard A. Frey, Steven A. Ackerman, Yinghui Liu, Kathleen I. Strabala, Hong Zhang, Jeffrey R. Key, and Xuangi Wang, “Cloud detection with modis. part i: Improvements in the modis cloud mask for collection 5,” *Journal of Atmospheric & Oceanic Technology*, vol. 25, no. 7, pp. 1057–1072, 2008.

[4] R Wl Saunders and K Ts Kriebel, “An improved method for detecting clear sky and cloudy radiances from avhrr data,” *International Journal of Remote Sensing*, vol. 9, no. 1, pp. 123–150, 1988.

[5] Martijn de Ruyter de Wildt, Gabriela Seiz, and Armin Gruen, “Operational snow mapping using multitemporal meteosat seviri imagery,” *Remote Sensing of Environment*, vol. 109, no. 1, pp. 29–41, 2007.

[6] Stanley Q Kidder, J Adam Kankiewicz, and Kenneth E Eis, “Meteosat second generation cloud algorithms for use at afwa,” *BACIMO, October*, pp. 12–14, 2005.

[7] Bo-Cai Gao, Ping Yang, and Rong-Rong Li, “Detection of high clouds in polar regions during the daytime using the modis 1.375-$\mu$m channel,” *IEEE transactions on geoscience and remote sensing*, vol. 41, no. 2, pp. 474–481, 2003.

[8] Farid Melgani and Lorenzo Bruzzone, “Classification of hyperspectral remote sensing images with support vector machines,” *IEEE Transactions on geoscience and remote sensing*, vol. 42, no. 8, pp. 1778–1790, 2004.

[9] Gustavo Camps-Valls and Lorenzo Bruzzone, “Kernel-based methods for hyperspectral image classification,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 6, pp. 1351–1362, 2005.

[10] Yoonkyung Lee, Grace Wahba, and Steven A Ackerman, “Cloud classification of satellite radiance data by multicategory support vector machines,” *Journal of Atmospheric and Oceanic Technology*, vol. 21, no. 2, pp. 159–169, 2004.

[11] MR Azimi-Sadjadi and SA Zekavat, “Cloud classification using support vector machines,” in *Geoscience and Remote Sensing Symposium, 2000. Proceedings. IGARSS 2000. IEEE 2000 International*. IEEE, 2000, vol. 2, pp. 669–671.

[12] Zhenyu An and Zhenwei Shi, “Scene learning for cloud detection on remote-sensing images,” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 8, pp. 4206–4222, 2015.

[13] Vijay Badrinarayanan, Ankur Handa, and Roberto Cipolla, “Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling,” *arXiv preprint arXiv:1505.07293*, 2015.

[14] Mengyun Shi, Fengying Xie, Yue Zi, and Jihao Yin, “Cloud detection of remote sensing images by deep learning,” in *Geoscience and Remote Sensing Symposium (IGARSS), 2016 IEEE International*. IEEE, 2016, pp. 701–704.

[15] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk, “Slic superpixels compared to state-of-the-art superpixel methods,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 11, pp. 2274–2282, 2012.

[16] Jonathan Long, Evan Shelhamer, and Trevor Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[17] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han, “Learning deconvolution network for semantic segmentation,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1520–1528.

[18] Sergey Ioffe and Christian Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *arXiv preprint arXiv:1502.03167*, 2015.

[19] Vinod Nair and Geoffrey E Hinton, “Rectified linear units improve restricted boltzmann machines,” in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 807–814.

[20] M Pawan Kumar, Benjamin Packer, and Daphne Koller, “Self-paced learning for latent variable models,” in *Advances in Neural Information Processing Systems*, 2010, pp. 1189–1197.

[21] Kevin Tang, Vignesh Ramanathan, Li Fei-Fei, and Daphne Koller, “Shifting weights: Adapting object detectors from image to video,” in *Advances in Neural Information Processing Systems*, 2012, pp. 638–646.

[22] François Chollet, “Keras,” `https://github.com/fchollet/keras`, 2015.

[23] Wan-Ting Lin, Chuen-Horng Lin, Tsung-Ho Wu, and Yung-Kuan Chan, “Image segmentation using the k-means algorithm for texture features,” *World Academy of Science, Engineering and Technology*, vol. 65, pp. 612–615, 2010.

[24] Dorin Comaniciu and Peter Meer, “Mean shift: A robust approach toward feature space analysis,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 5, pp. 603–619, 2002.