

ACLNet: An Attention and Clustering-based Cloud Segmentation Network

Dhruv Makwana¹ and Subhrajit Nag² and Onkar Susladkar³ and Gayatri Deshmukh³ and Sai Chandra Teja R¹ and Sparsh Mittal⁴ and C Krishna Mohan^{1,2}

¹CKM VIGIL Pvt Ltd, Hyderabad, India; ²CSE Department, IIT Hyderabad, India; ³CSE Department, Vishwakarma Institute of Information Technology, Pune, India; ⁴Mehta Family School of Data Science and Artificial Intelligence, IIT Roorkee, India

ARTICLE HISTORY

Compiled July 14, 2022

ABSTRACT

We propose a novel deep learning model named ACLNet, for cloud segmentation from ground images. ACLNet uses both deep neural network and machine learning (ML) algorithm to extract complementary features. Specifically, it uses EfficientNet-B0 as the backbone, “à trous spatial pyramid pooling” (ASPP) to learn at multiple receptive fields, and “global attention module” (GAM) to extract fine-grained details from the image. ACLNet also uses k -means clustering to extract cloud boundaries more precisely. ACLNet is effective for both daytime and nighttime images. It provides lower error rate, higher recall and higher F1-score than state-of-art cloud segmentation models. The source-code of ACLNet is available here: <https://github.com/ckmvigil/ACLNet>.

KEYWORDS

Cloud segmentation; attention; k -means clustering; day and night images

1. Introduction

Cloud segmentation has several applications such as weather prediction, climate hazard prediction, and solar energy forecasting. A study of clouds can provide crucial clues about the hydrological scorecard of the atmosphere and impending climate hazards. Further, clouds hamper satellite cameras’ views and affect solar energy generation plants. Driven by these factors, the detection of clouds has gained a lot of traction in recent years. However, an accurate segmentation of clouds has remained a challenge due to factors such as the non-rigid shape of clouds and variable lighting especially of nighttime images.

Previous works have performed cloud segmentation on images taken from satellites (Xie et al. 2017), and those taken from the ground (Dev, Lee, and Winkler 2016; Shi et al. 2020; Xie et al. 2020). We review those works that analyze cloud images taken from ground. Previous works use color information to distinguish cloud (blue) from

CONTACT C Krishna Mohan. Email: ckm@cse.iith.ac.in. Dhruv and Subhrajit are co-first authors. This work was supported by WNI WxBunka Foundation, Japan. The computing systems used in this research were provided by Indian Institute of Technology, Roorkee, India under grant FIG-100874.

the sky (white). For example, in the technique of Long, Shelhamer, and Darrell (2015), the cloud area is detected by setting a threshold value for the ratio of the pixel values of red and blue. However, this method assumes that the colors of the sky and clouds in the daytime can be clearly seen. But this is not effective for images with a small difference in color between the sky and clouds, such as images at night, dawn, or dusk.

Shi et al. (2019) propose a VGG16-based fully-convolutional network. They incorporate “histogram equalization” and “skip connections” to achieve effective segmentation. Xie et al. (2020) use an encoder-decoder architecture. They modify the VGG16 network by replacing the fully connected layers with the decoder network. “CloudU-Net” (Shi et al. 2020) is inspired from U-Net and it uses “dilated convolutions” and “fully connected conditional random field (CRF)”. It also uses a lookahead optimizer for faster model convergence. “CloudSegNet” (Dev et al. 2019) is an encoder-decoder architecture and has been evaluated on both daytime and nighttime images. Table 1 shows the key characteristics of a few related works.

Table 1. Key characteristics of methods used in recent studies (SLIC = simple linear iterative clustering, TCDD = TJNU Cloud Detection Database, TLCDD = TJNU Large-scale Cloud Detection Database)

	Architecture and highlights	Dataset
CloudSegNet (Dev et al. 2019)	Encoder-decoder CNN with Conv and DeConv layers	SWIMSEG, SWINSEG
Dev, Lee, and Winkler (2014)	Fuzzy c -means clustering	HYTA
Dev et al. (2017)	SLIC for superpixel generation and k -means clustering	SWINSEG
Xie et al. (2017)	SLIC for superpixel generation and CNN for classifying superpixels into thick/thin-cloud and sky. CNN has 2 branches to extract features at two scales	Quickbird satellite, Google map, internet images
CloudU-Net (Shi et al. 2020)	Encoder-decoder CNN based on U-Net. Fully-connected CRF layers, dilated Conv	GDNCI
MACNN (Zhang et al. 2021)	Encoder-decoder CNN with attention. Dilated Conv with different dilation values to capture multiscale information	TCDD
CloudRaednet (Shi, Zhou, and Qiu 2022)	Residual attention-based encoder-decoder CNN. ResNet50 as encoder; residual modules in decoder; Ranger optimizer	GDNCI
DPNet (Zhang et al. 2022)	Encoder-decoder CNN. Spatial pyramid pooling and then fusing the features by attention weights	TLCDD

Further, several previous works such as “CRF as recurrent neural networks” (CRF-RNN) have large computation and model-size overheads.

In this paper, we propose a DNN for segmenting clouds from both daytime and nighttime sky images. ACLNet utilizes EfficientNet-B0 (Tan and Le 2019) as the backbone. This allows ACLNet to learn better feature maps since EfficientNet-B0 is a highly regularized model. Also, we use the “à trous spatial pyramid pooling” (ASPP) module (Chen et al. 2018) to learn at multiple receptive fields. This helps in detecting clouds of different sizes. Further, ACLNet uses two key novelties. First, we propose a “global attention module” (GAM) to focus on regions of interest, strengthen the learning of channels, and improve training efficiency. Second, we propose using k -means clustering for detecting cloud boundaries. This helps in learning features that complement those learnt by the DNN. We combine this information with the DNN-extracted features for generating the final segmentation mask.

ACLNet has lower error rate, higher recall and higher F1-score than the state-of-art models, for daytime, nighttime, and day+night time images (corresponding to SWIMSEG, SWINSEG and SWINySEG datasets, respectively). The main contributions of the paper are: (1) An interpretable and accurate framework targeted for cloud segmentation from both daytime and nighttime images. (2) We introduce an ML algorithm

in our DNN to bring their best together and more effectively segment the clouds.

2. ACLNet: Proposed network

ACLNet is a novel network that outputs a cloud segmentation binary mask. Figure 1 depicts the proposed ACLNet architecture. Here, a Conv block consists of a convolution layer followed by a batchnorm and ReLU activation. ACLNet uses EfficientNet-B0 (Tan and Le 2019) as the backbone network. EfficientNet-B0 is a highly regularised model. It has been obtained by performing a neural architecture search and jointly optimizing FLOPS, and accuracy. We now describe other components of ACLNet.

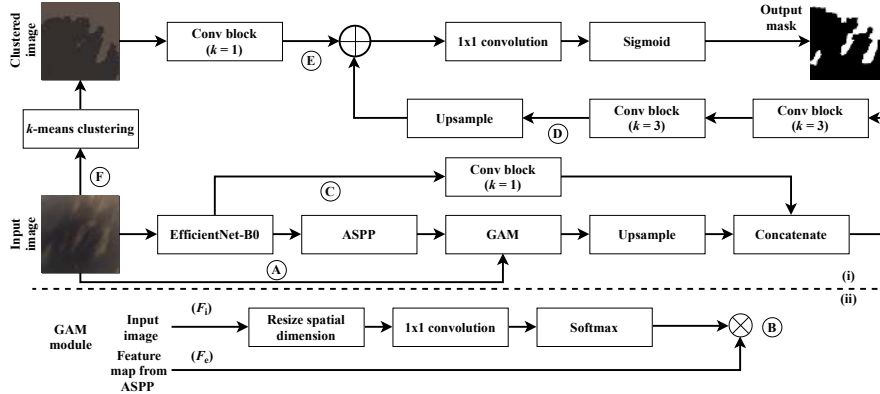


Figure 1. (i) ACLNet architecture (k denotes the kernel size) (ii) GAM module

2.1. ASPP Module

Clouds vary widely in their size. To effectively detect clouds of various sizes, we seek to resample the features of a single scale. For this purpose, we use ASPP block (Chen et al. 2018). The input to ASPP is the downsampled feature map derived from the final layer of the backbone. ASPP uses parallel à trous convolution layers with different sampling rates, processes these features in different branches, and fuses them. Specifically, ASPP has an average pooling layer with global information features, a Conv block with kernel size of 1×1 for extracting original scale features, and three Conv blocks having kernel size of 3×3 with dilation rates of 6, 12, and 18, respectively. Thus, it uses multiple receptive fields. Finally, ASPP concatenates the feature maps and uses a Conv block with 1×1 kernel size to reduce the number of channels.

While extracting multi-scale features, ASPP substantially reduces the information loss caused by numerous downsampling operations. It assesses convolutional features at several scales by applying à trous convolution with different dilation rates to image-level features. It reduces the number of parameters due to the use of three dilated convolutions. Furthermore, to maintain more location information, the sizes of distinct feature maps at different scales are kept the same.

2.2. Global Attention Module (GAM)

Motivation: We need an attention mechanism for two reasons. (1) The output from ASPP has little semantic information, but the input image provides rich semantic

information that may be utilized to aid a low-level output from the ASPP feature in capturing semantic dependencies. Therefore, improving the contextual information of the low-level output from the ASPP feature makes it easier to ensure effectual fusion.

(2) The input image contains rich contextual information. However, the contextual information provided by the ASPP output is inadequate for pixel-wise recognition. To compensate for this loss of contextual information, one can concatenate the original input image with the ASPP output. However, concatenating two incompatible features negatively impacts the semantic gaps between the ASPP output and the corresponding input image. We need to include additional semantic information into the ASPP output to fuse them effectively.

We introduce GAM to achieve above two objectives. GAM adaptively boosts location and semantic information by giving pixel-level and channel-level attention to the image. The design of GAM is shown in Figure 1(ii).

Working of GAM: Let ASPP output be $F_e \in \mathbb{R}^{H_1 \times W_1 \times C_1}$, where H_1 and W_1 are height and width of the feature map and C_1 is the number of channels. Similarly, let input image be $F_i \in \mathbb{R}^{H_2 \times W_2 \times C_2}$. In GAM, we proceed as follows:

(1) Spatially resize the image (refer ① in Figure 1) to match the spatial dimension of ASPP output. The image-dimension becomes $H_1 \times W_1 \times C_2$. (2) Use 1×1 convolution to capture channel dependencies for creating squeeze channel attention map of size $H_1 \times W_1 \times C_1$. (3) Apply a softmax activation on the channel attention map to obtain the pixel-wise attention weights. This generates the attention map $A \in \mathbb{R}^{H_1 \times W_1 \times C_1}$. (4) To create the final refined features, multiply the above attention map with ASPP output feature map F_e . This produces the output shown as ② in Figure 1.

The output of GAM is up-sampled to refine the features. We concatenate this up-sampled feature map with the matching low-level feature map from the network backbone, which has the same spatial resolution. Before this concatenation, we pass the low-level feature map obtained from the backbone network (refer ③ in Figure 1) through a Conv block with kernel size of 1×1 to reduce the number of channels. This is because the low-level features typically contain a large number of channels, which can outweigh the importance of the output from ASPP features.

2.3. Using clustered image

Recent cloud-segmentation techniques exclusively use DNN. However, as we show in Section 4, the use of a DNN alone does not provide high predictive performance because a DNN learns by considering features that are independent of each other. Conventional ML algorithms can identify how different types of data are interrelated and create new segments based on those relationships. To bring the best of both worlds together, we use clustering along with feature maps learned by DNN. This helps in finding the relationship between the features that can be segmented.

To efficiently generate binary masks, we need information about cloud boundaries. To obtain cloud boundaries, we apply k -means clustering on the RGB pixel values of the input image (refer ④ in Figure 1). This clusters similar pixels together while creating a border between sky and cloud. We use a centroid value of two to create two clusters, viz., a cloud region, and a non-cloud region. We want to learn different features from cloud and sky from daytime and nighttime images. Hence, we have used k -means and not edge detection algorithms like Sobel edge detector.

Following concatenation, the resultant feature map is processed through two Conv blocks having kernel size 3×3 to enhance the features (refer ⑤ in Figure 1). Then,

bilinear upsampling is performed. The clustered input image is passed through Conv block with kernel size 1 (refer ⑤ in Figure 1) and then added with the upsampled feature map obtained above. Finally, this feature map is passed through a 1×1 convolution layer, reducing the number of channels to two. This output is passed through sigmoid activation to generate the output mask.

Overall network design: Starting from ② in Fig. 1, we first up-sample the refined features. We concatenate this upsampled feature map with the matching low-level feature map from the network backbone, which has the same spatial resolution. Before this concatenation, we pass the low-level feature map obtained from the backbone network (refer ③ in Figure 1) through a Conv block with kernel size of 1×1 to reduce the number of channels. This is because the low-level features typically contain a large number of channels, which can outweigh the importance of the output from ASPP features.

Following concatenation, the resultant feature map is processed through two Conv blocks having kernel size 3×3 to enhance the features (refer ④ in Figure 1). Then, bilinear upsampling is performed. The clustered input image is passed through Conv block with a kernel size of 1×1 (refer ⑤ in Figure 1) and then added with the upsampled feature map obtained above. Finally, this feature map is passed through a 1×1 convolution layer, reducing the number of channels to two. This output is passed through sigmoid activation to generate the output mask.

3. Experimental Platform

Dataset: We have used SWINySEG (Dev et al. 2019) dataset for training our model, which has images of size 300×300 pixels. SWINySEG is a composite dataset consisting of augmentations applied to the SWIMSEG (Dev, Lee, and Winkler 2016) and SWINSEG (Dev et al. 2017) dataset to create a balance between daytime and nighttime images. We use random sampling to divide the training and testing sets in an 80:20 ratio on this composite dataset. The SWINySEG dataset has images consisting of a combination of images taken both during daytime and nighttime. The SWIMSEG dataset has 1013 images taken during the daytime, while on the other hand, SWINSEG has 115 images taken during the nighttime. The SWIMSEG dataset has 1013 images taken during the daytime, while on the other hand, SWINSEG has 115 images taken during the nighttime. During the inference phase, we resize the image to 300×300 pixels and then use a random crop.

Training settings: We use TensorFlow 2.6. We use the Adam optimizer with an initial learning rate of 0.0001. Whenever there is no convergence for 20 epochs continuously, the learning rate is dynamically varied for fine-tuning. We use a batch size of 8 to train the model. We perform end-to-end training of ACLNet to minimize segmentation loss. For segmentation, we use BinaryCrossEntropy-DICE (BCE-DICE) loss. BCE-DICE loss is a combination of the distribution-based loss function (BCE) and region-based loss function (DICE loss). BCE loss considers each pixel as an independent prediction and optimizes loss in under-segmented regions. DICE optimizes loss in over-segmented regions. The training is done for 300 epochs. The dice score saturates after 270 epochs. Because loss is always convergent towards global minima, the network’s training is consistent overall. We observe that use of k -means clustering has little impact on the training and testing time of the network.

Evaluation metrics: We use (1) precision, (2) recall, (3) F1-score, and (4) error rate defined as $\frac{FP+FN}{TP+FP+TN+FN}$. Here, FP means false positive, TP means true positive, FN

means false negative and TN means true negative. (5) mean intersection over union (MIoU), defined as $MIoU = \frac{1}{|C|} \sum_{c \in C} J_c(y^*, \tilde{y})$ where y^* and \tilde{y} contain the ground truth and predicted labels of all pixels in the testing dataset. C denotes all the classes and c denotes individual classes from C . J_c is the Jaccard index of class c . (6) ‘‘Matthews correlation coefficient’’ (MCC), defined as $MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$. (7) We show the ROC (receiver operating characteristic) curve, which depicts classifier performance at all classification thresholds. We have performed three trials of experiments with ACLNet and observed that the standard deviation of results across different trials is negligibly small.

4. Results

Visualization: Figure 2 (i)-(v) shows five sample images (three daytime and two nighttime images) from the SWINySEG dataset. It also compares the ground truth masks (b) with the output masks produced by ACLNet (c), U-Net (d) and DeepLabv3+ (e). We can see that the output density map generated by ACLNet is quite similar to the ground-truth density estimation map. ACLNet preserves the cloud boundaries, and by virtue of using the clustering, it preserves the pixel information for generating a binary mask. For all these images, ACLNet leads to more accurate pixel-count than U-Net and DeepLabv3+. In (v), none of the models give good predicted pixel count, still ACLNet performs much better than the other models.

Quantitative results: We compare ACLNet with five other baseline networks, viz., FCN, CloudU-Net, DeepLabv3+, U-Net and CloudSegNet. Note that we have ourselves trained these networks and performed experiments to obtain the results. Table 2 shows the experimental results. Evidently, ACLNet provides the best value of metrics for all cases, except for precision metric on nighttime images.

Table 2. Results on daytime, nighttime and day+night time images (Best values are shown in **bold font**)

Method	Precision	Recall	F1-Score	Error Rate	MIoU	MCC
Daytime (SWIMSEG)						
FCN	0.532	0.466	0.456	0.502	0.651	0.724
CloudU-Net	0.951	0.971	0.952	0.042	0.963	0.853
DeepLabv3+	0.889	0.913	0.888	0.082	0.971	0.93
U-Net	0.771	0.772	0.754	0.191	0.870	0.812
CloudSegNet	0.921	0.897	0.892	0.078	0.944	0.826
CloudSegNet (with clustering)	0.941	0.914	0.912	0.061	0.955	0.885
ACLNet (proposed)	0.964	0.979	0.971	0.022	0.992	0.956
Nighttime (SWINSEG)						
FCN	0.423	0.492	0.431	0.567	0.591	0.681
CloudU-Net	0.943	0.951	0.941	0.049	0.931	0.816
DeepLabv3+	0.864	0.962	0.891	0.084	0.961	0.901
U-Net	0.693	0.673	0.701	0.240	0.842	0.782
CloudSegNet	0.891	0.924	0.881	0.083	0.915	0.824
CloudSegNet (with clustering)	0.932	0.934	0.901	0.075	0.931	0.861
ACLNet (proposed)	0.917	0.982	0.947	0.037	0.985	0.930
Day + Night Time (SWINySEG)						
FCN	0.500	0.511	0.441	0.555	0.591	0.713
CloudU-Net	0.956	0.967	0.952	0.044	0.941	0.945
DeepLabv3+	0.861	0.906	0.852	0.084	0.973	0.93
U-Net	0.714	0.764	0.741	0.216	0.855	0.8
CloudSegNet	0.930	0.883	0.891	0.079	0.926	0.812
CloudSegNet (with clustering)	0.931	0.943	0.922	0.071	0.932	0.873
ACLNet (proposed)	0.959	0.979	0.968	0.024	0.993	0.960


























	(a) Original Image	(b) Original Mask	(c) ACLNet Predicted Mask	(d) U-Net Predicted Mask	(e) DeepLabv3+ Predicted Mask
(i)		 39878	 39736 (-142)	 37089 (-2789)	 41634 (+1756)
(ii)		 44654	 44663 (+9)	 46109 (+1455)	 43899 (-755)
(iii)		 28704	 28208 (-496)	 35124 (+6420)	 31702 (+2988)
(iv)		 28805	 29309 (+504)	 32921 (+4116)	 33703 (+4898)
(v)		 17424	 22427 (+5003)	 24674 (+7250)	 28286 (+10862)

Figure 2. Segmentation results for three daytime (i), (ii),(v) and two nighttime (iii)-(iv) images. The number of pixels in the cloud mask is shown below each figure. The red-color text in parenthesis shows the difference in pixel-count between the ground-truth and the prediction made by a model (ACLNet/U-Net/DeepLabV3+).

The error rate quantifies the rate of pixel-wise misclassification between ground truth and predicted output over the whole set of instances. Error rate measures the inaccuracy of predicted output values for target values. ACLNet achieves the lowest error-rate for all three categories. These results confirm the superiority of our model.

Based on the histogram peak of daytime and nighttime images, the difference between the color of sky and cloud regions is better in daytime images than in nighttime images. Hence, ACLNet performs well on daytime images but produces slightly inferior results on nighttime images. Although ACLNet’s precision on the nighttime images is slightly less, the cloud features are efficiently segmented.

In this paper, our key idea is that since machine-learning (ML) and deep-learning (DL) networks extract complementary features, combining them can provide higher performance. Our ACLNet network validates this idea and for further validation, we evaluate combining k -means clustering with CloudSegNet network. On comparing “CloudSegNet” results with “CloudSegNet (with clustering)” in Table 2, it is clear that k -means clustering helps in improving the predictive performance of CloudSegNet also. This confirms our design-choice and the importance of our innovation.

ROC curve: Figure 3 compares the ROC curves of ACLNet with the other baseline networks shown in Table 2. The area under the curve (AUC) of ROC for CloudU-Net

is the least while the AUC-ROC curve for ACLNet is the highest. The higher the true positive rate and lower the false positive rate, the higher is the AUC. This shows that ACLNet has the highest true positive rate while having the lowest false positive rate.

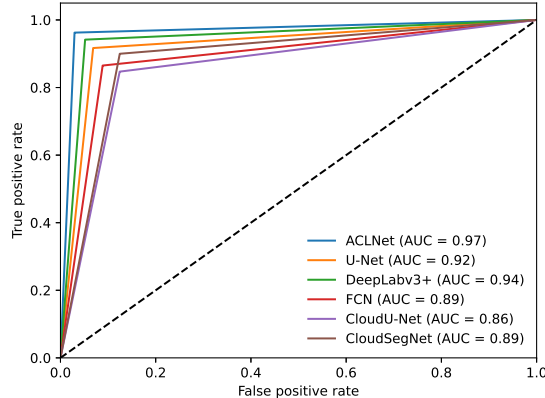


Figure 3. ROC curve comparison of different segmentation models.

Model size and throughput: From Table 3, we can see that the model size of ACLNet is lower than all techniques, except CloudSegNet. While CloudSegNet has a small model size, but its predictive performance is much inferior compared to ACLNet. CloudU-Net has a very high model size (138MB). Overall, ACLNet achieves the best balance of model size and performance. Also, ACLNet requires only 7.57 million FLOPs. ACLNet has a throughput of 5.7 and 13.1 frames-per-second on 2080Ti GPU and P100 GPU, respectively.

Table 3. Model size (MB) of different cloud segmentation models

Model	Size	Model	Size
CloudU-Net	138.6	CloudSegNet	0.11
DeepLabv3+	61.3	ACLNet with EfficientNet-B1 backbone	36.7
U-Net	94.3	ACLNet with EfficientNet-B2 backbone	43.1
FCN (Long, Shelhamer, and Darrell 2015)	53.6	ACLNet with ViT backbone	325
ACLNet with ResNet50 backbone	47.1	ACLNet	30.79

Ablation studies:

1. Impact of removing GAM and k -means: We now evaluate the contribution of GAM and k -means by removing them individually and together. Table 4 shows the results. In Figure 1, on removing GAM, the path shown with (A)-(B) is removed. On removing k -means clustering, the path shown with (F)-(E) in Figure 1 is removed. On removing the k -means clustering, all metrics become worse. Clearly, the features extracted by DNN are not sufficient for achieving high predictive performance for cloud segmentation. On removing the GAM, all metrics, especially the precision for nighttime images, become worse. ACLNet without k -means gives inferior results as compared to ACLNet without GAM in precision and F1-Score for nighttime images, whereas the results are almost equal for the daytime and day+night images. On removing both k -means and GAM, the results become even worse (results omitted for brevity). Thus, both k -means and attention modules are important for achieving high performance.

2. Impact of changing the backbone: Table 5 shows the results of replacing the backbone in ACLNet to four other networks: EfficientNet-B1 and B2, ViT and ResNet50. We observe that for all these four backbones, the predictive performance does not increase much compared to that with EfficientNet-B0. Further, on using ViT, the model

Table 4. Results of ablation studies on GAM and k -means clustering (P=precision, R=recall, ER=error-rate)

	ACLNet without GAM						ACLNet without k -means					
	P	R	F1	ER	MIoU	MCC	P	R	F1	ER	MIoU	MCC
Day	0.94	0.93	0.94	0.05	0.97	0.91	0.92	0.95	0.94	0.073	0.96	0.918
Night	0.82	0.97	0.88	0.078	0.95	0.87	0.91	0.92	0.93	0.078	0.95	0.895
Day + night	0.94	0.92	0.94	0.05	0.98	0.9	0.93	0.94	0.93	0.074	0.96	0.913

size increases to 325 MB. This is much larger than the model size with EfficientNet-B0 backbone. On 2080Ti GPU and P100 GPU, ACLNet (with EfficientNet-B0 backbone) have throughput of 5.7 and 13.1 frames-per-second, respectively. For ACLNet (with ViT backbone), these numbers are 1.8 and 5.0, respectively. Thus, use of EfficientNet-B0 leads to higher throughput than use of ViT. Considering predictive performance, model size and throughput, we prefer EfficientNet-B0 as the backbone.

Table 5. Results of ablation studies on ACLNet with different backbones (P=precision, R=recall, ER=error-rate)

	ACLNet with EfficientNet-B1 backbone						ACLNet with EfficientNet-B2 backbone					
	P	R	F1	ER	MIoU	MCC	P	R	F1	ER	MIoU	MCC
Day	0.96	0.98	0.97	0.022	0.98	0.95	0.98	0.98	0.98	0.022	0.98	0.95
Night	0.91	0.98	0.95	0.033	0.98	0.94	0.92	0.98	0.96	0.033	0.97	0.94
Day + night	0.96	0.98	0.97	0.023	0.97	0.95	0.96	0.98	0.98	0.022	0.99	0.95
	ACLNet with ViT backbone						ACLNet with ResNet50 backbone					
	P	R	F1	ER	MIoU	MCC	P	R	F1	ER	MIoU	MCC
Day	0.98	0.98	0.98	0.022	0.99	0.95	0.96	0.98	0.96	0.025	0.98	0.95
Night	0.94	0.97	0.95	0.031	0.98	0.94	0.92	0.98	0.95	0.037	0.96	0.93
Day + night	0.97	0.98	0.97	0.022	0.99	0.95	0.96	0.98	0.97	0.026	0.98	0.95

3. Use of k -means++ clustering: k -means++ (Arthur and Vassilvitskii 2006) acts as an intelligent initialization procedure for k -means. k -means++ algorithm minimizes the likelihood of a wrong initialization but has higher computational overhead. However, we observe that k -means++ algorithm provides no improvement over the k -means algorithm. This is because of the use of a smaller K value (two), such that random or intelligent initialization has a negligible impact.

5. Conclusion and Future Work

We present a novel model, ACLNet, that can accurately segment clouds from both nocturnal and daytime images. ACLNet achieves high predictive performance by virtue of combining the benefits of “à trous spatial pyramid pooling”, attention and clustering. ACLNet is lightweight and has the highest recall and F1-score for all types of images.

ACLNet is trained to work on the predetermined classes - namely cloud and sky. However, a real-world image may also have other objects such as birds or plane. ACLNet may not work well with such unseen classes. To achieve better performance on those unseen classes, we would need to retrain or fine-tune our model. In near future, we plan to train ACLNet using unsupervised learning to further improve its segmentation performance. Vision transformer models provide state-of-art results on several computer vision tasks such as classification (Liu et al. 2021) and segmentation (Strudel et al. 2021; Xie et al. 2021; Zheng et al. 2021). Our future work will focus on comprehensively evaluating transformer-based models for cloud segmentation.

References

- Arthur, David, and Sergei Vassilvitskii. 2006. *K-means++: The advantages of careful seeding*. Technical Report. Stanford.
- Chen, Liang-Chieh, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. 2018. “Encoder-decoder with atrous separable convolution for semantic image segmentation.” In *European Conference on Computer Vision (ECCV)*, 801–818.
- Dev, Soumyabrata, Yee Hui Lee, and Stefan Winkler. 2014. “Systematic study of color spaces and components for the segmentation of sky/cloud images.” In *2014 IEEE International Conference on Image Processing (ICIP)*, 5102–5106. IEEE.
- Dev, Soumyabrata, Yee Hui Lee, and Stefan Winkler. 2016. “Color-based segmentation of sky/cloud images from ground-based cameras.” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing (J-STARS)* 10 (1): 231–242.
- Dev, Soumyabrata, Atul Nautiyal, Yee Hui Lee, and Stefan Winkler. 2019. “Cloudsegnet: A deep network for nychthemeron cloud image segmentation.” *IEEE Geoscience and Remote Sensing Letters* 16 (12): 1814–1818.
- Dev, Soumyabrata, Florian M Savoy, Yee Hui Lee, and Stefan Winkler. 2017. “Nighttime sky/cloud image segmentation.” In *IEEE International Conference on Image Processing (ICIP)*, 345–349. IEEE.
- Liu, Ze, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. “Swin transformer: Hierarchical vision transformer using shifted windows.” In *International Conference on Computer Vision*, 10012–10022.
- Long, Jonathan, Evan Shelhamer, and Trevor Darrell. 2015. “Fully convolutional networks for semantic segmentation.” In *Proceedings of the IEEE CVPR*, 3431–3440.
- Shi, Chaojun, Yatong Zhou, and Bo Qiu. 2022. “CloudRaednet: residual attention-based encoder–decoder network for ground-based cloud images segmentation in nychthemeron.” *International Journal of Remote Sensing* 43 (6): 2059–2075.
- Shi, Chaojun, Yatong Zhou, Bo Qiu, Dongjiao Guo, and Mengci Li. 2020. “CloudU-Net: A Deep Convolutional Neural Network Architecture for Daytime and Nighttime Cloud Images’ Segmentation.” *IEEE Geoscience and Remote Sensing Letters* 18 (10): 1688–1692.
- Shi, Chaojun, Yatong Zhou, Bo Qiu, Jingfei He, Mu Ding, and Shiya Wei. 2019. “Diurnal and nocturnal cloud segmentation of all-sky imager (ASI) images using enhancement fully convolutional networks.” *Atmospheric Measurement Techniques* 12 (9): 4713–4724.
- Strudel, Robin, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. 2021. “Segmenter: Transformer for semantic segmentation.” In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 7262–7272.
- Tan, Mingxing, and Quoc Le. 2019. “Efficientnet: Rethinking model scaling for convolutional neural networks.” In *International conference on machine learning*, 6105–6114.
- Xie, Enze, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. 2021. “SegFormer: Simple and efficient design for semantic segmentation with transformers.” *Advances in Neural Information Processing Systems* 34: 12077–12090.
- Xie, Fengying, Mengyun Shi, Zhenwei Shi, Jihao Yin, and Danpei Zhao. 2017. “Multilevel cloud detection in remote sensing images based on deep learning.” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 10 (8): 3631–3640.
- Xie, Wanyi, Dong Liu, Ming Yang, Shaoqing Chen, Benge Wang, Zhenzhu Wang, Yingwei Xia, Yong Liu, Yiren Wang, and Chaofang Zhang. 2020. “SegCloud: a novel cloud image segmentation model using a deep convolutional neural network for ground-based all-sky-view camera observation.” *Atmospheric Measurement Techniques* 13 (4): 1953–1961.
- Zhang, Zhong, Shuzhen Yang, Shuang Liu, Xiaozhong Cao, and Tariq S Durrani. 2022. “Ground-based Remote Sensing Cloud Detection using Dual Pyramid Network and Encoder-Decoder Constraint.” *IEEE Transactions on Geoscience and Remote Sensing* 60.
- Zhang, Zhong, Shuzhen Yang, Shuang Liu, Baihua Xiao, and Xiaozhong Cao. 2021. “Ground-Based Cloud Detection Using Multiscale Attention Convolutional Neural Network.” *IEEE Geoscience and Remote Sensing Letters* 19: 1–5.

Zheng, Sixiao, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, et al. 2021. “Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers.” In *IEEE/CVF conference on computer vision and pattern recognition*, 6881–6890.