Noname manuscript No.
(will be inserted by the editor)

U-Net based Sky segmentation utilizing K-means cluster with transient attributes trained XGBoost

Vardan Agarwal*, Yohan Varghese Kuriakose*,
Rahul Dixit

the date of receipt and acceptance should be inserted later

Abstract In this paper, we present a new algorithm for sky segmentation which is based on clustering and the UNET segmentation model. Color spaces like RGB, YCrCb, and HSV are used to train a clustering model, an XGBoost model trained on transient attributes which are high-level features and is used to classify new images into the clusters defined by the clustering model. Finally, for each cluster, a UNET segmentation model is trained for classification into sky or non-sky pixel and to generate a mask. Experimental results show that our model performs well with the SkyFinder dataset achieving a mean accuracy of 96.7% and a mean IOU of 93.1% on the testing set. Further testing with ADE20K dataset, only using images containing sky pixels, an accuracy of 94.9% was achieved.

1 Introduction

Image Segmentation [2] is the process of classifying pixels belonging to an object, in this case, the sky. Sky Segmentation is a complex and challenging process in real-world scenarios because of the non-rigid horizon between sky and non-sky pixels. With the variety of geographical conditions like cities, towns, and forests, and weather conditions it becomes difficult for baseline models to classify.

To solve these problems, we used clustering to divide the dataset into clusters by using the color models RGB, HSV, and YCbCr [11]. The mean values of only the sky pixels are calculated for each of the constituents of these color-spaces and used as an input for K-means clustering [14]. To predict which cluster any new image would be placed in is

Name of First Author
Vardan Agarwal*, Department of Computer Science and Engineering, Manipal University Jaipur, E-mail: vardanagarwal16@gmail.com

Name of Second Author
Yohan Varghese Kuriakose*, Department of Computer Science and Engineering, Manipal University Jaipur, E-mail: yohanvarghese9655@gmail.com

Name of Third Author
Rahul Dixit, Department of Computer Science and Engineering, Indian Institute of Information Technology Pune, E-mail: rahul2012ism@gmail.com
*Both authors are lead authors and have contributed equally

achieved by using transient attributes introduced by Laffont et al. [12] which are 40 high-level features of an outdoor scene that can be predicted for any image. An XGBoost classifier [3] trained on these features is used to classify new images into different clusters. Finally, for segmentation, separate fully convolutional networks [15,24] are trained for each cluster which will finally classify pixels as sky or non-sky and generate a binary mask.

Sky segmentation has various uses like in image dehazing [17,25] to identify the atmospheric light which is needed to develop transmission maps. It can also be used in sky cloud segmentation [7,6] which is used for forecasting weather, predicting rainfalls and other meteorological activities and is currently performed on images with only the sky. Using sky segmentation, we can identify the sky regions and apply cloud segmentation in only those areas. It also has an application in UAVs for obstacle avoidance [5,18], image editing [21] and weather estimation [4].

Over the years many researchers have tried to solve the problem of sky segmentation. Mihail et al. [19] stressed upon the lack of real-world datasets for sky segmentation and that the benchmark datasets available had images captured in favorable conditions. They presented a new dataset SkyFinder and evaluated some baseline models of Hoeim et al. [8], Tighe et al. [23] and Lu et al. [16] on it. They also presented a new deep learning-based ensemble model for sky segmentation. Hoeim et al. attempted to divide images into three categories: ground, vertical and sky, by learning appearance-based models of geometric classes from a single image. Tighe et al. used scene-level matching with global image descriptors followed by superpixel-level matching with local features and efficient Markov random field. Through this, they were able to label images for objects and geometric classes, i.e. sky, vertical, and ground. Lu et al. classified the images into two categories: sunny and cloudy. They used random forests and graphs cut to perform sky segmentation. Yazdanpanah et al. [27] proposed a sky segmentation model by fusing K-means clustering and neural network classifications. This is used in automated path planning and obstacle avoidance of NASA's Mars rover. Place et al. [20] evaluated off the shelf models for sky segmentation as well as fine-tuned and trained RefineNet-ResNet models from scratch. They also investigated the effects of time and weather on the model.

1.1 Our contribution

We introduce a novel method to divide the dataset into clusters based on the sky pixels for different color spaces. This reduces the variation of the sky and makes it easier for the models to learn about the characteristics of the sky. To predict which cluster any unseen image would lie in only transient attributes are used. No abstract details like time, temperature, humidity, etc are included to ensure that our model is not compromised due to non-availability of data. We also tested our data on ADE20K dataset [29] to see our model performs in the real world. Other than we also see how our method performs in different weather conditions.

The rest of this paper is organized as follows. We discuss the proposed method in detail in Section 2. This is followed by experimental results in Section 3. Comparison with other techniques is provided in Section 4. Finally, the concluding remarks and future worth looking for work is provided in Section 5.

2 Proposed Method

In this section, we present the proposed method for sky segmentation. The method can be divided into three broad categories, which are described in detail, next: (a) Clustering using K-means; (b) XGBoost classifier for predicting clusters using transient attributes; (c) Segmentation models for each cluster.

2.1 Clustering Using K-Means

The images which were initially in RGB format are converted to both HSV and YCbCr formats. The HSV format is a cylindrical color model and is closer to how humans perceive color while the YCbCr color model is a practical approximation to color processing and perceptual uniformity. The equations used to convert RGB to HSV are provided from equation (1) to (6) and for RGB to YCbCr are from equation (7) to (9). For each image, the mean values of all the sky pixels are calculated for all the nine matrices (R, G, B, H, S, V, Y, Cb, Cr). These values are passed as input to the K-means algorithm.

The K-means algorithm divides the data points into predefined k non-overlapping clusters. It performs the grouping procedure by calculating the sum of squared distance between the centroid of a cluster and a given data point. The value of k is calculated using the elbow method by plotting the values for distortion and inertia for different values of k. Distortion is the average of the squared Euclidean distances from the cluster centers of the respective clusters. Inertia is calculated as the sum of squared distances of samples to their closest cluster center. The values obtained are shown in Fig. 1. From the figure it was clear that the elbow is formed at k=2, so the clustering algorithm was trained with a value of k=2. The clustering data is reduced to two dimensions using Principal Component Analysis (PCA) [26]. The divided dataset is shown in Fig. 2. On inspection, it can be seen that the first cluster contains images of the day with light sky and the second cluster contains images of the night with dark sky.

$$R' = \frac{R}{255}, G' = \frac{G}{255}, B' = \frac{B}{255} \tag{1}$$

$$Cmax = max(R', G', B'), Cmin = min(R', G', B') \tag{2}$$

$$\Delta = C_{max} - C_{min} \tag{3}$$

$$H = \begin{cases} 0^\circ, \Delta = 0 \\ 60^\circ \times \left(\frac{G'-B'}{\Delta} mod 6 \right), C_{max} = R' \\ 60^\circ \times \left(\frac{B'-R'}{\Delta} + 2 \right), C_{max} = G' \\ 60^\circ \times \left(\frac{R'-G'}{\Delta} + 4 \right), C_{max} = B' \end{cases} \tag{4}$$

$$S = \begin{cases} 0, C_{max} = 0 \\ \frac{\Delta}{C_{max}}, C_{max} \neq 0 \end{cases} \tag{5}$$

$$V = Cmax \tag{6}$$

$$Y = 16 + \frac{65.738R}{256} + \frac{129.057G}{256} + \frac{25.064B}{256} \tag{7}$$

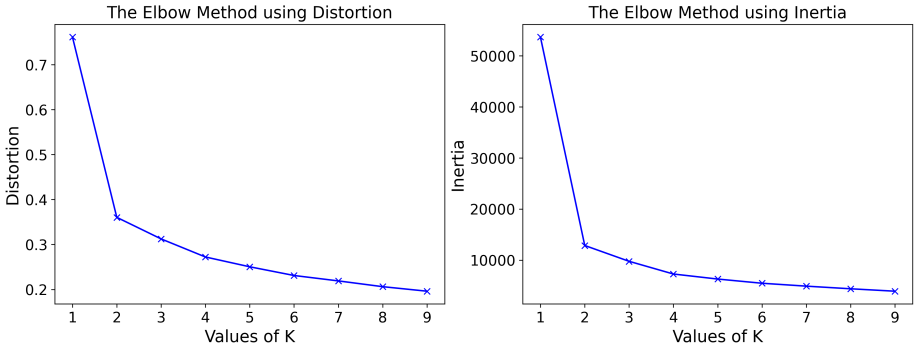


Fig. 1: Elbow Method to find k

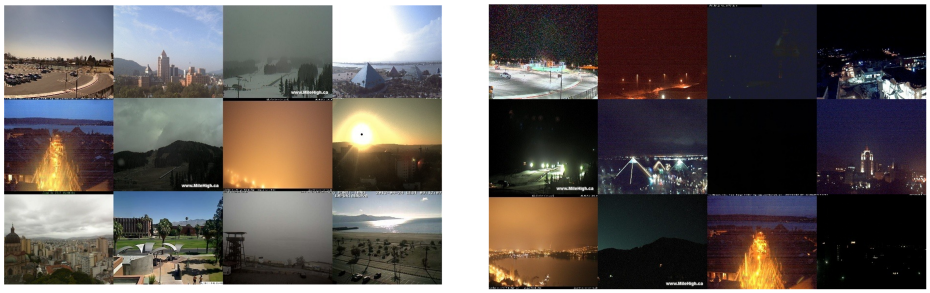


Fig. 2: Dataset divided in different clusters

$$Cb = 128 - \frac{37.945R}{256} - \frac{74.494G}{256} + \frac{112.439B}{256} \tag{8}$$

$$Cr = 128 + \frac{112.439R}{256} - \frac{94.154G}{256} - \frac{18.285B}{256} \tag{9}$$

2.2 XGBoost Classifier for Predicting Clusters Using Transient Attributes

Predicting the cluster in which any new image would lie in was not possible by using the K-means clustering model because in that only the sky pixels were used. Therefore, we used transient attributes presented by Laffont et al. [12] for predicting which cluster should any test image lie in using XGBoost classifier.

Boosting refers to train a family of algorithms or models to convert weak learners to strong learners. Each model is trained sequentially and try correcting the predecessor's errors. XGBoost is an ensemble machine learning algorithm using decision trees as its base and gradient boosting framework. It uses algorithms like Parallelization, Tree Pruning, Regularization (Lasso L1 and Ridge L2), Sparsity Awareness, weighted quantile sketch, and cross-validation to provide system optimization and better results. The XGBoost classifier was trained with a maximum depth of 3 and the number of estimators set to 1000. Clustering resulted in unbalanced classes and the number of samples was more in set 0 than set 1. To

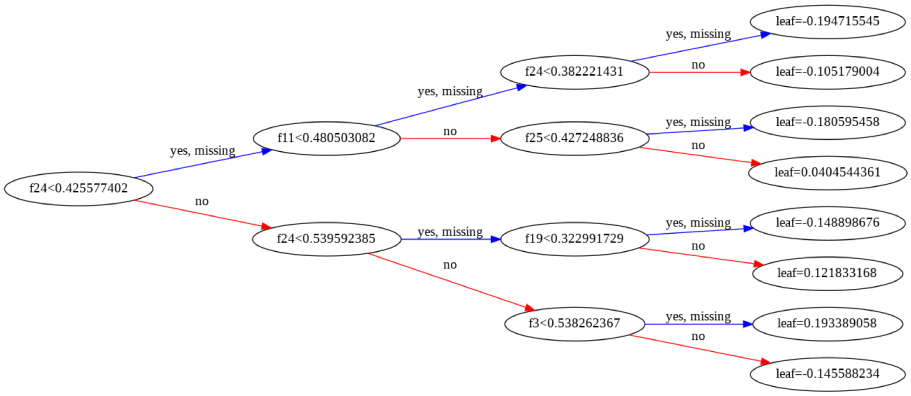


Fig. 3: Xgboost Tree

counter its negative effects, the values of scale positive weight was set to 1.6. Other than this subsampling ratio was set to 0.5 and the initial learning rate to 0.1. The XGBoost tree generated is shown in Fig. 3

2.3 Segmentation Models for each Cluster

For segmentation, we created a fully convolutional network in Tensorflow [1]. In a fully convolutional network, there are no Dense layers. The pooling layers are replaced by upsampling operations also known as deconvolutional layers. These layers are used to increase the resolution of the output. There are many such architectures like UNet [22], Pyramid Scene Parsing Network (PSPNet) [28], Feature Pyramid Networks (FPN) [13], etc. which implement fully convolutional networks. For our method, we use the UNet architecture which was developed by Ronnenberger et al. [22] for biomedical image segmentation. The network consists of a contracting path where spatial information is reduced and feature information is increased and an expansive path which consists of a large number of feature channels for precise localization of objects.

As shown by Place et al. [20] we also trained our network from scratch to utilize the enormous size of the dataset. The size of the dataset is reduced to 224×224 to speed up training and evaluation purposes. Image augmentation operations like horizontal flip and random crop followed by resizing were performed to help the model generalize better and prevent overfitting. Layers from MobileNetV3 [9] (block_1_expand_relu, block_3_expand_relu, block_6_expand_relu, block_13_expand_relu, block_16_project) were selected for the purpose of downsampling. All upsampling layers except the last one had a dropout of 0.5 and the number of feature channels used were 512, 256, 128, and 64 in order. The model created has a total of 6,502,786 parameters out of which 6,469,954 are trainable. The model architecture is shown below in Fig. 4. The loss function used is sparse categorical cross-entropy and the optimizer is set to Adam [10]. The model was trained for 6 epochs with an initial learning rate of 1×10^{-4}. Reduce learning rate on plateau was used as a callback to reduce the learning rate whenever the model stagnates with a patience of 2. This allows the model to better reach the minima without constantly overshooting it. The flowchart of the method is shown in Fig. 5

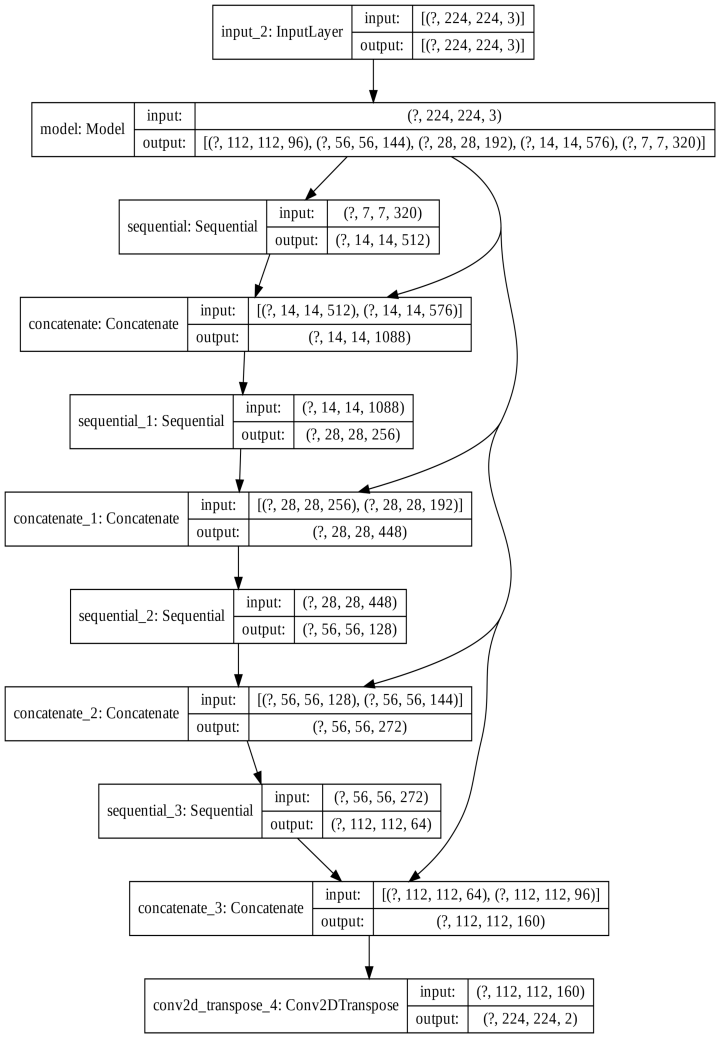


Fig. 4: Model Architecture

Algorithm of Proposed Method

Input: SkyFinder Dataset $images, mask$, attributes att

1. Let $image_{train}$, att_{train}, $image_{test}$, att_{test}, $image_{val}$, att_{val} denote the variables used to denote images and attributes of train, test and validation set and let $skyatt_{train}$ denote the pixel attributes of the training set.

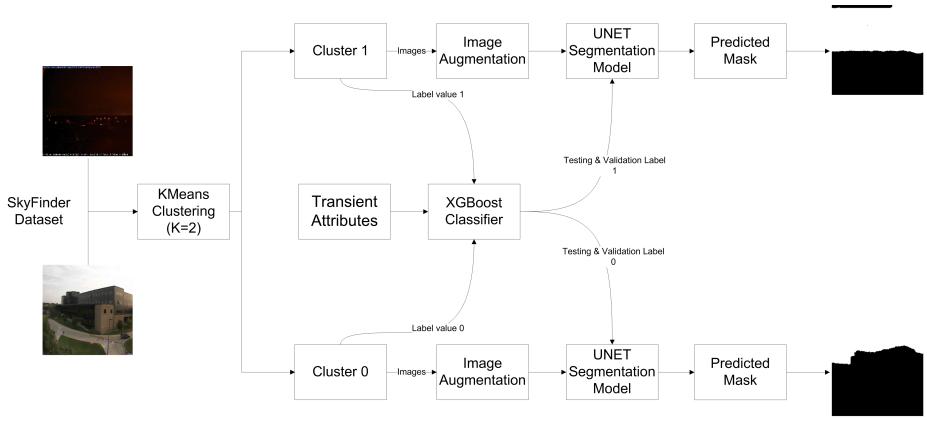


Fig. 5: Flow Chart

2. Let images and attributes of 6 CamId′s be removed from $image_{train}$, att_{train}, $image_{val}$, att_{val} and added to test set $image_{test}$ and att_{test}.

3. Split images data set *images* and its attributes *att* into $image_{train}$, att_{train}, $image_{val}$ and att_{val} with 80-20% split.

4. Initialize *KMeans* with KMeans Cluster with number of clusters = 2.

5. Fit KMeans cluster model *Kmeans* with attributes of training set $skyatt_{train}$ to get $labels_{train}$ as output containing 0 or 1.

6. Using XGBoost algorithm for classification, fit with att_{train} as input and $labels_{train}$ as actual output.

7. Let fit XGBoost model be used to predict output of att_{val} and store it in $labels_{val}$.

8. Let fit XGBoost model be used to predict output of att_{test} and store it in $labels_{test}$.

9. Split $images_{train}$ into clusters using $labels_{train}$ as reference to get $cluster0_{train}$ and $cluster1_{train}$.

10. Split $images_{test}$ and $images_{val}$ using $label_{test}$ and $label_{val}$ to get $cluster0_{test}$, $cluster1_{test}$, $cluster0_{val}$ and $cluster1_{val}$.

11. Initialize $Model0$ and $Model1$ with Unet segmentation algorithm.

12. Let *batchSize* and *epoch* denote the batch size of the dataset to be used for training and the number of epochs in training.

13. Initialize data generators $dataGen0_{train}$, $dataGen1_{train}$, $dataGen0_{val}$, $dataGen1_{val}$ for images $cluster0_{train}$, $cluster1_{train}$, $cluster0_{val}$ and $cluster1_{val}$ respectively with pre-processing steps like cropping and horizontal flipping.

14. Train $Model0$ and $Model1$ with $dataGen0_{train}$ and $dataGen1_{train}$ as input data generators, *mask* as actual output and $dataGen0_{val}$ and $dataGen1_{val}$ as validation data generators to be used after each epoch.

15. Test models $Model0$ and $Model1$ with $cluster0_{test}$ and $cluster1_{test}$.

3 Experimental Results

The proposed method was tested with the SkyFinder dataset which contains 77992 images captured from different web cameras. Out of these 77992 images, only 77769 are usable rest are corrupted. The dataset is divided into different "Cam-Ids" which denotes the different regions where the image was captured. Each Cam-Id contains the images of the same location albeit in different conditions like time, weather etc. The dataset contains 47 different CamId's i.e 47 different locations. Out of this 6 randomly selected CamIDs (75, 1093, 5021, 8438, 10917, 19834) were taken as the testing set which in conjunction have 9248 images. The other CamIds were split into 80%-20% for training and validation purposes. The metrics used for the analysis are accuracy and intersection over union (IOU) and are shown in equations (10) and (11) respectively.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{10}$$

$$IntersectionoverUnion = \frac{TP}{TP+FP+FN} \tag{11}$$

where, TP denotes true positives (correctly determined sky pixels), TN denotes true negatives (correctly determined non-sky pixels), FP denotes false positives and FN denotes false negatives.

Figure 6 shows the clustering data reduced to two dimensions using Principal Component Analysis (PCA) when k=2. As one can see the two clusters have well defined non overlapping regions. XGBoost algorithm was used to train the attributes to get the output 0 or 1. The trained model gave an accuracy of 99.2% for the validation set.

Figure 7 and 8 respectively shows the training and validation performance against epochs and learning rate. The performance is measured with accuracy, loss, Intersection Over Union (IOU). Similarly, figure 9 and 10 shows the training and validation performance against epochs and learning rate respectively, again measured with the same metrics and loss. From 8 and 10 we can see that the learning rate was at 1.0×10^{-4} for the first 3 epochs thereafter once the rate of change of the metrics started to reduce, the learning rate was reduced to 1.0×10^{-6} to observe its effect on the results, which was a further improvement in the metrics and loss.

Table 1 shows the performance of the test set. As one can see CamId 75 label 1 gives less than satisfactory results, this is due to some images being totally black due to the absence of lights in the scene. CamId 10917 doesn't have any sky region within its images hence the mean IOU value is 0 or not applicable. Some sample predictions for cluster 0 and cluster 1 are in shown in figures 11 and 12 respectively.

We also tested our method on the training and validation set images from ADE20K Dataset which had a combined total of 22210 images. It contained 150 segmentation categories like the sky, road, grass, etc. but we tested it for only sky and non-sky. For testing,

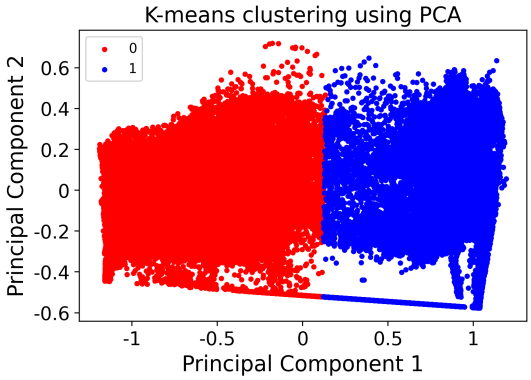


Fig. 6: Visualizing K-means clustering using PCA

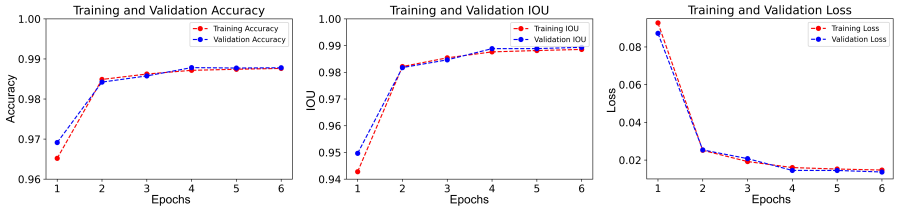


Fig. 7: Epoch Vs Performance of Training and Validation set Cluster 0

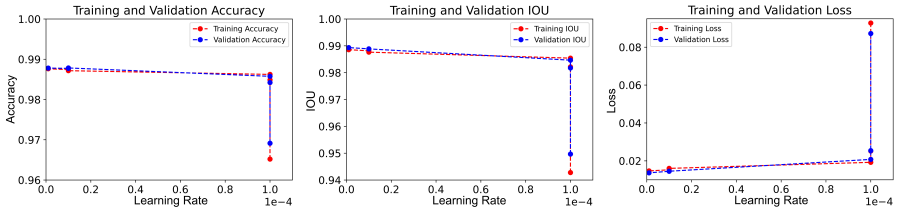


Fig. 8: Learning Rate Vs Performance of Training and Validation set Cluster 0

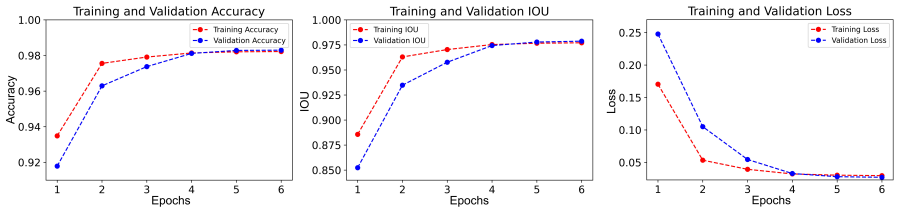


Fig. 9: Epoch Vs Performance of Training and Validation set Cluster 1

only those images were selected which had at least one sky pixel. From the images containing sky pixels, we achieved an accuracy of 94.9% and an mIOU of 69.8%. The reason for the low mIOU value is because many of the selected images in this data set contain a very

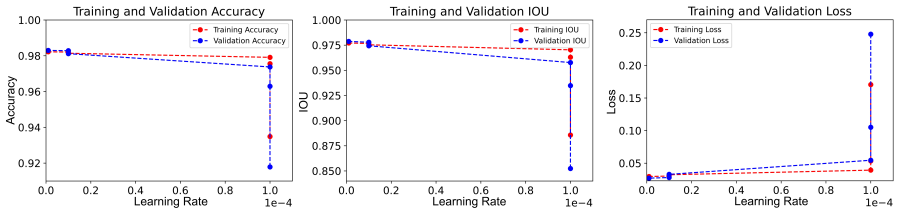


Fig. 10: Learning Rate Vs Performance of Training and Validation set Cluster 1

Table 1: Performance of Test Set

CamId	label	Accuracy	mean IOU
75	0	0.958	0.949
75	1	0.733	0.678
1093	0	0.978	0.930
1093	1	0.943	0.821
5021	0	0.977	0.950
5021	1	0.976	0.949
8438	0	0.977	0.947
8438	1	0.959	0.906
10917	0	1.000	-
10917	1	0.908	-
19834	0	0.976	0.955
19834	1	0.966	0.937

small percentage of sky pixels which means the number of true positives is very less. After removing those images having less than 10% sky pixel the mIOU value improved to 82%. Some predictions are shown in Fig. 13.

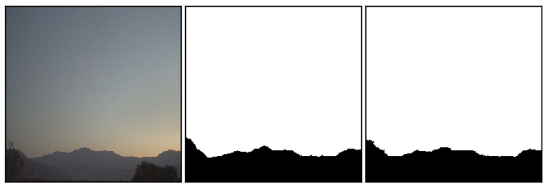
4 Comparison with other methods

To test the performance of our method we have compared it to 3 other baseline methods proposed by Hoeim et al. Tighe et al. and Lu et al based on Misclassification Rate (MCR) and is calculated as shown in equation (12). We could not evaluate our method against Mihail et al. who did not provide which CamIds were used for testing and reported an overall MCR 12.96%. Table 2 shows the comparison.

$$MCR = \frac{FP + FN}{TP + TN + FP + FN} \tag{12}$$

where FP, FN, TP, and TN denote False Positives, False Negatives, True Positives, and True Negatives respectively.

Table 3 shows the comparison of our method and others for different weather conditions again calculated based on MCR. As one can see, our method performs better in most weather conditions.

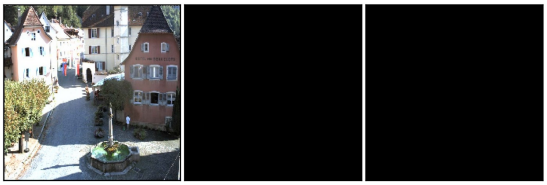


(a) Original Image (b) Ground truth (c) Output Image

(d) Original Image (e) Ground truth (f) Output Image

(g) Original Image (h) Ground truth (i) Output Image

(j) Original Image (k) Ground truth (l) Output Image



(m) Original Image (n) Ground truth (o) Output Image

(p) Original Image (q) Ground truth (r) Output Image

Fig. 11: Cluster 0 outputs

(a) Original Image (b) Ground truth (c) Output Image

(d) Original Image (e) Ground truth (f) Output Image

(g) Original Image (h) Ground truth (i) Output Image

(j) Original Image (k) Ground truth (l) Output Image

(m) Original Image (n) Ground truth (o) Output Image

(p) Original Image (q) Ground truth (r) Output Image

Fig. 12: Cluster 1 outputs

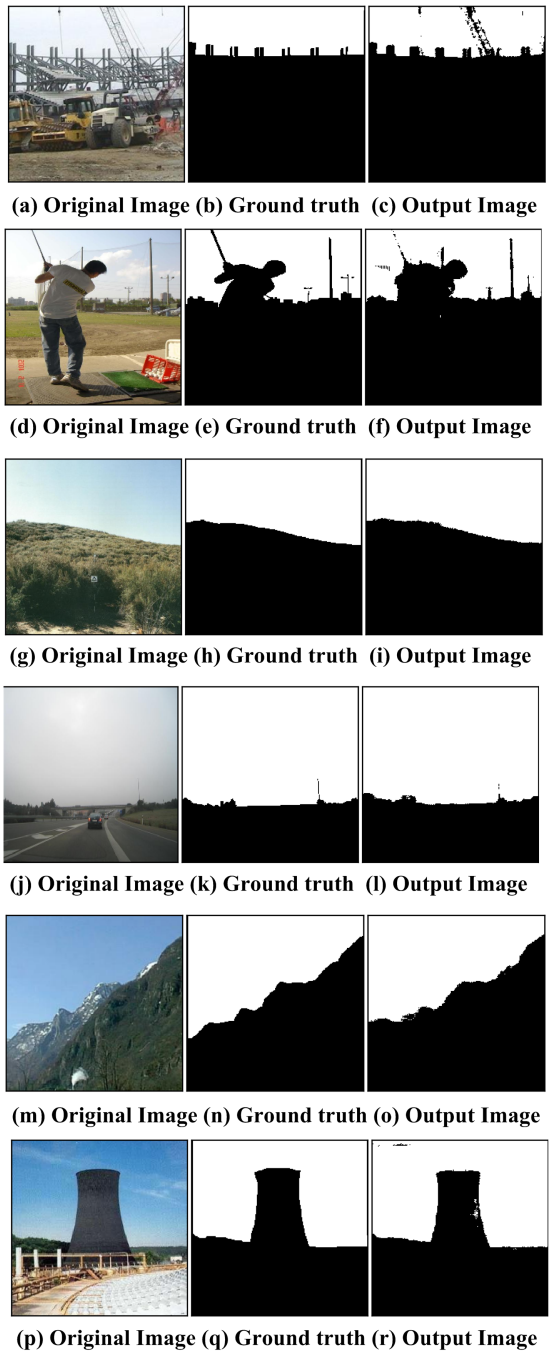


Fig. 13: ADE20k dataset outputs

Table 2: Comparative analysis of different methods

CamId	label	Hoiem et al.	Tighe et al.	Lu et al.	Our Method
75	0	0.098	0.050	0.829	0.042
75	1	0.420	0.130	0.830	0.267
1093	0	0.112	0.051	0.034	0.022
1093	1	0.364	0.273	0.338	0.057
5021	0	0.127	0.072	0.086	0.023
5021	1	0.176	0.207	0.450	0.024
8438	0	0.136	0.039	0.114	0.023
8438	1	0.457	0.499	0.438	0.041
10917	0	0.254	0.577	0.008	0.000
10917	1	0.177	0.725	0.004	0.092
19834	0	0.148	0.035	0.475	0.024
19834	1	0.272	0.412	0.530	0.034

Table 3: Comparative analysis of different methods in different weather condition

Icon	Hoiem et al.	Tighe et al.	Lu et al.	Our Method
clear	0.185	0.263	0.420	0.039
cloudy	0.222	0.307	0.216	0.032
fog	0.231	0.400	0.200	0.044
hazy	0.198	0.221	0.415	0.044
mostly cloudy	0.205	0.377	0.177	0.025
partly cloudy	0.210	0.312	0.233	0.026
rain	0.213	0.486	0.093	0.026
sleet	0.248	0.515	0.003	0.040
snow	0.260	0.538	0.028	0.031
thunder storms	0.193	0.218	0.281	0.030
unknown	0.199	0.475	0.144	0.028

5 Conclusion

In this paper, we propose a sky segmentation technique where instead of directly applying a fully convolutional network on the dataset we clustered the images based on the sky pixels. These clusters were then predicted using transient attributes for any new image using an XGBoost classifier. Finally, fully convolutional networks based on the UNet architecture were trained for each cluster. Our experimental results show that our model performs much better than baseline models which shows that off the shelf methods don't work properly for this task.

For future work, we would like to try different architectures of fully convolutional networks like FPN, PSPNET, etc. individually on all clusters to see if they can improve the final result.

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mane, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viegas, F.,

Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: Tensorflow: Large-scale machine learning on heterogeneous distributed systems (2016)
2. Badrinarayanan, V., Kendall, A., Cipolla, R.: Segnet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **39**(12), 2481–2495 (2017)
3. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, p. 785–794. Association for Computing Machinery, New York, NY, USA (2016). DOI 10.1145/2939672.2939785
4. Chu, W.T., Zheng, X.Y., Ding, D.S.: Camera as weather sensor: Estimating weather information from single images. Journal of Visual Communication and Image Representation **46**, 233 – 249 (2017). DOI https://doi.org/10.1016/j.jvcir.2017.04.002
5. de Croon, G.C.H.E., De Wagter, C., Remes, B.D.W., Ruijsink, R.: Sky segmentation approach to obstacle avoidance. In: 2011 Aerospace Conference, pp. 1–16 (2011)
6. Dev, S., Lee, Y.H., Winkler, S.: Color-based segmentation of sky/cloud images from ground-based cameras. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing **10**(1), 231–242 (2017)
7. Dev, S., Manandhar, S., Lee, Y.H., Winkler, S.: Multi-label cloud segmentation using a deep network. In: 2019 USNC-URSI Radio Science Meeting (Joint with AP-S Symposium), pp. 113–114 (2019)
8. Hoiem, D., Efros, A.A., Hebert, M.: Geometric context from a single image. In: Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1, vol. 1, pp. 654–661 Vol. 1 (2005)
9. Howard, A., Sandler, M., Chu, G., Chen, L.C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., Le, Q.V., Adam, H.: Searching for mobilenetv3. In: The IEEE International Conference on Computer Vision (ICCV) (2019)
10. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2014)
11. Kolkur, S., Kalbande, D., Shimpi, P., Bapat, C., Jatakia, J.: Human skin detection using rgb, hsv and ycbcr color models. Proceedings of the International Conference on Communication and Signal Processing 2016 (ICCASP 2016) (2017). DOI 10.2991/iccasp-16.2017.51
12. Laffont, P.Y., Ren, Z., Tao, X., Qian, C., Hays, J.: Transient attributes for high-level understanding and editing of outdoor scenes. ACM Transactions on Graphics (proceedings of SIGGRAPH) **33**(4) (2014)
13. Liang, Z., Shao, J., Zhang, D., Gao, L.: Small object detection using deep feature pyramid networks. In: R. Hong, W.H. Cheng, T. Yamasaki, M. Wang, C.W. Ngo (eds.) Advances in Multimedia Information Processing – PCM 2018, pp. 554–564. Springer International Publishing, Cham (2018)
14. Likas, A., Vlassis, N., Verbeek], J.J.: The global k-means clustering algorithm. Pattern Recognition **36**(2), 451 – 461 (2003). DOI https://doi.org/10.1016/S0031-3203(02)00060-2. Biometrics
15. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)
16. Lu, C., Lin, D., Jia, J., Tang, C.: Two-class weather classification. IEEE Transactions on Pattern Analysis and Machine Intelligence **39**(12), 2510–2524 (2017)
17. Ma, Q., Pei, T., Wang, X., Tang, Q., Zhou, Q., Yu, T.: Single image dehazing using sky adaptive fusion. In: 2019 IEEE 4th International Conference on Image, Vision and Computing (ICIVC), pp. 208–212 (2019)
18. Mashaly, A.S., Wang, Y., Liu, Q.: Efficient sky segmentation approach for small uav autonomous obstacles avoidance in cluttered environment. In: 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 6710–6713 (2016)
19. Mihail, R.P., Workman, S., Bessinger, Z., Jacobs, N.: Sky segmentation in the wild: An empirical study. In: IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 1–6 (2016)
20. Place, C.L., Urooj, A., Borji, A.: Segmenting sky pixels in images: Analysis and comparison. In: 2019 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 1734–1742 (2019)
21. Rawat, S., Gairola, S., Shah, R., Narayanan, P.J.: Find me a sky: A data-driven method for color-consistent sky search and replacement. In: K. Schoeffmann, T.H. Chalidabhongse, C.W. Ngo, S. Aramvith, N.E. O'Connor, Y.S. Ho, M. Gabbouj, A. Elgammal (eds.) MultiMedia Modeling, pp. 216–228. Springer International Publishing, Cham (2018)
22. Ronneberger, O., P.Fischer, Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Medical Image Computing and Computer-Assisted Intervention (MICCAI), *LNCS*, vol. 9351, pp. 234–241. Springer (2015). (available on arXiv:1505.04597 [cs.CV])
23. Tighe, J., Lazebnik, S.: Superparsing: Scalable nonparametric image parsing with superpixels. In: K. Daniilidis, P. Maragos, N. Paragios (eds.) Computer Vision – ECCV 2010, pp. 352–365. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)
24. Wang, L., Ouyang, W., Wang, X., Lu, H.: Visual tracking with fully convolutional networks. In: The IEEE International Conference on Computer Vision (ICCV) (2015)

25. Wang, W., Yuan, X., Wu, X., Liu, Y.: Dehazing for images with large sky region. Neurocomputing **238**, 365 – 376 (2017). DOI https://doi.org/10.1016/j.neucom.2017.01.075
26. Wold, S., Esbensen, K., Geladi, P.: Principal component analysis. Chemometrics and Intelligent Laboratory Systems **2**(1), 37 – 52 (1987). DOI https://doi.org/10.1016/0169-7439(87)80084-9. Proceedings of the Multivariate Statistical Workshop for Geologists and Geochemists
27. Yazdanpanah, A.P., Regentova, E.E., Mandava, A.K., Ahmad, T., Bebis, G.: Sky segmentation by fusing clustering with neural networks. In: G. Bebis, R. Boyle, B. Parvin, D. Koracin, B. Li, F. Porikli, V. Zordan, J. Klosowski, S. Coquillart, X. Luo, M. Chen, D. Gotz (eds.) Advances in Visual Computing, pp. 663–672. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)
28. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6230–6239 (2017)
29. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ade20k dataset. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2017)