ClusterNet: Unsupervised Generic Feature Learning for Fast Interactive Satellite Image Segmentation

Nicolas Girard, Andrii Zhygallo, Yuliya Tarabalka

ClusterNet: Unsupervised Generic Feature Learning for Fast Interactive Satellite Image Segmentation

Nicolas Girard[a], Andrii Zhygallo[a], and Yuliya Tarabalka[a,b]

[a]Université Côte d'Azur, Inria
[b]LuxCarta Technology

ABSTRACT

Semantic segmentation on satellite images is used to automatically detect and classify objects of interest over very large areas. Training a neural network for this task generally requires a lot of human-made ground truth classification masks for each object class of interest. We aim to reduce the time spent by humans in the whole process of image segmentation by learning generic features in an unsupervised manner. Those features are then used to leverage sparse human annotations to compute a dense segmentation of the image. This is achieved by essentially labeling groups of semantically similar pixels at once, instead of labeling each pixel almost individually using strokes. While we apply this method to satellite images, our approach is generic and can be applied to any image and to any class of objects on that image.

Keywords: Deep Learning, Unsupervised, Clustering, Satellite Image, Segmentation, Interactive

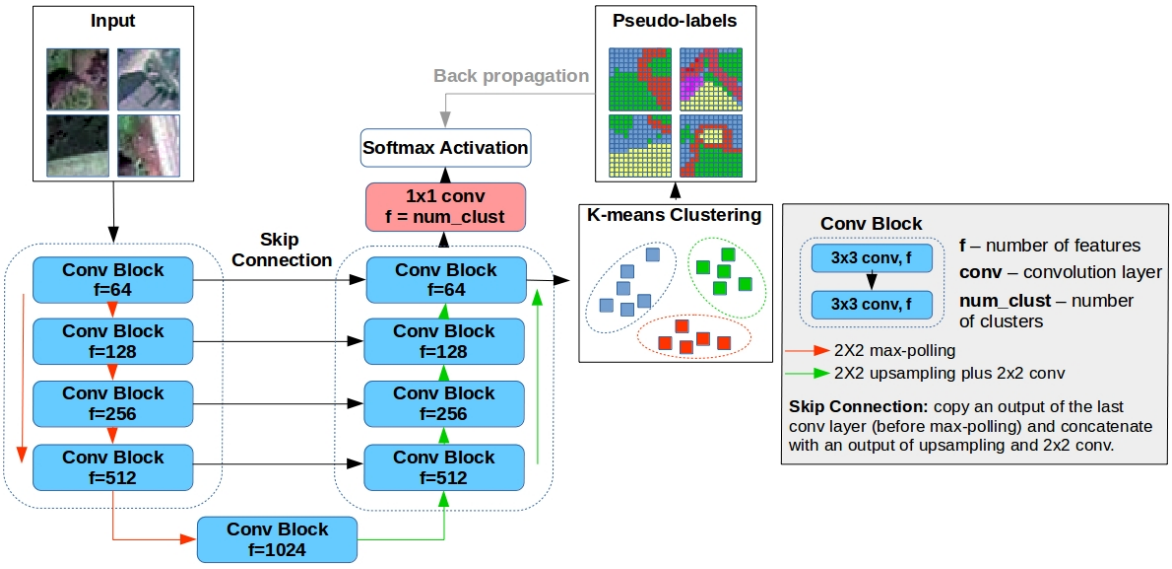


Figure 1: First step of our ClusterNet method: image inputs go through a U-Net[1] network that outputs f=64 features per pixel. Those features are then fed first to a k-means clustering algorithm that assigns for each pixel a cluster, which is now viewed as a pseudo-label. In addition, the output features are also fed to a fully-connected layer (1x1 convolution) + softmax which outputs a classification vector of size num_clust per pixel. The cross-entropy loss is used on the predicted labels and the pseudo-labels.

Nicolas Girard: nicolas.girard@inria.fr, Yuliya Tarabalka: ytarabalka@luxcarta.com

1. INTRODUCTION

Overhead images cover the whole earth, with petabytes of image data being captured every day.[2] Training an instance segmentation model for these images requires a very big and diverse dataset because of the rich visual variations of the landscapes and human constructions around the globe. Gathering such massive ground truth data is a very time consuming endeavor. Additionally, even if some ground truth exists, it is often not precise and has errors of its own (see Fig.2). Reducing the amount and/or detail of ground truth needed to train for a task has been the goal of new kinds of machine learning paradigms such as k-shot learning, weakly-supervision, semi-supervised and unsupervised learning.

K-shot learning tackles the task of building a model that is capable of learning a new class with very few examples. In the extreme case no example is provided, leading to zero-shot learning. These methods aim to reproduce the human ability to learn a new concept from very few examples. For a survey of k-shot learning methods, see "Few-shot Learning: A Survey"[3] .

Weakly-supervision aims to use ground truth data of a lower meaning level in order to produce predictions at a higher level of meaning. For example, a ground truth label would be a labeled bounding box[4,5] around an object and the prediction would be a segmentation mask. Other examples for segmentation is to provide as ground truth a labeled scribble inside objects,[6] or a labeled point[7,8] or even only provide image-level labels.[9–11] Weakly-supervised methods work because they incorporate additional human knowledge and assumptions about the data in the form of a priors or constraints on the prediction. For example compactness (prediction must be around labeled point), inclusiveness (prediction must be inside the bounding box), prior shapes of different classes,using NLP on image-level labels, etc.

Semi-supervised methods use datasets for which only a typically small fraction of samples have a ground truth. Those methods still leverage the unlabeled data to learn more robust features for example. An extreme case of semi-supervision is unsupervised learning, where there is no ground truth. In unsupervised learning, the goal is to find structure in the data, most often in the form of clusters. One such clustering algorithm known as k-means is relatively effective for its simplicity. It has been used in conjunction to deep learning methods to provide pseudo-labels for a CNN,[12] which was our inspiration for our work. We used the feature learning with k-means method from[12] in a fully-convolutional manner to work with image segmentation. Then we developed a cluster classification method that leverages sparse human inputs in an interactive manner to classify each cluster. Another similar work is[13] which uses the JULE[14] (Join Unsupervised LEarning) framework with k-mans for medial 3D image segmentation. They however do not use a fully-convolutional approach, instead relying on patches extracted from the images. There method is also completely unsupervised as they use an agglomerative clustering method to reach the target number of clusters whereas we use some human supervision to classify each cluster in an interactive manner. Another approach to unsupervised image segmentation is W-Net[15] which uses an encoder-decoder network made out of 2 U-Net.[1] The latent representation of the image given by the first U-Net is further processed by a hierarchical segmentation method to output the segmentation. A reconstruction loss at the end of the network is used to train the encoder-decoder and a soft normalized cut loss is used on the segmentation output to train for clustering.

Our method is a combination of these learning paradigms. We use unsupervised deep learning to learn general features in a first step that are later used the second step using a weakly-supervised classification algorithm for quick interactive annotation.

Graph-cut methods are related to our second step. For example GrabCut[16] reduces the amount of user input for foreground extraction. Another method mixes SVM and Graph-cut[17] for hyperspectral image segmentation, using a SVM to predict the data term of the graph-cut optimization. Grap-cut methods partition the image in the spatial domain, making it faster to segment an image, however each separate object of the same class have to be marked by the user. In our approach we partition the pixels in a learned feature space allowing the user to mark just a few objects of the same class with a few strokes and all the other objects of the same class will be segmented as well. This makes segmentation of large satellite images a lot faster than graph-cut methods.

(a) Pittsburg (b) San Diego

Figure 2: Example of available imperfect ground truth for trees.

2. METHODOLOGY

The ClusterNet method is divided in 2 steps. The first is to learn for each pixel of the image a meaningful feature vector in an unsupervised manner (summarized in Fig.1). The second step is an interactive image segmentation which receives sparse human inputs in the form of labeled strokes.

2.1 Generic Unsupervised Feature Learning

Algorithm 1: Unsupervised feature learning

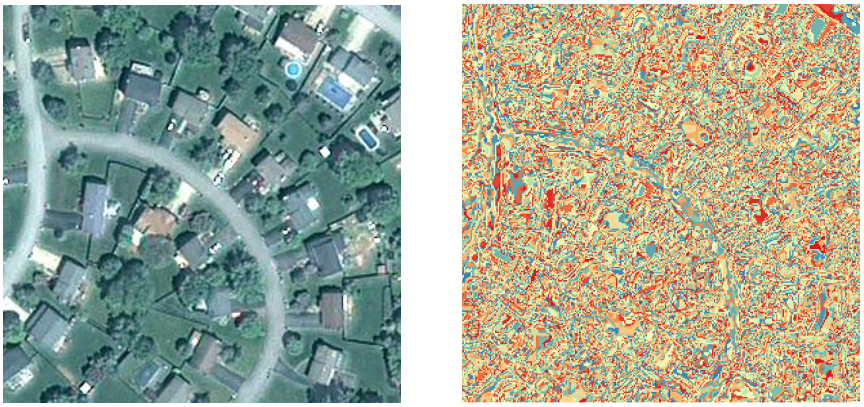
Input: Images $\mathcal{I} = \{I_1, ..., I_n\}$, number of epochs e, cluster batch size b_c, train batch size $b_t <= b_c$, number of clusters k

Initialize U-Net model M with random weights;
Initialize cluster centroids $\mathcal{K} = \{p_1, ..., p_k\}$;
for $i = 1$ **to** e **do**
  $\mathcal{B}_c = batch(\mathcal{I}, b_c)$; // Group images into clustering batches
  for B_c *in* $\mathcal{B}_c$ **do**
    // Compute cluster/class index for each pixel and update centroids $\mathcal{K}$:
    $\mathcal{L}_c, \mathcal{K} = kmeans(B_c, \mathcal{K})$; // $\mathcal{L}_c = \{L_1, ..., L_{b_c}; L_i$ image with values in $[0..k[\}$
    $\mathcal{B}_t, \mathcal{L}_t = batch([B_c, \mathcal{L}_c], b_t)$; // Group images and pseudo-labels into smaller batches
    for B_t, L_t *in* $\mathcal{B}_t, \mathcal{L}_t$ **do**
      $M = train(M, B_t, \mathcal{L}_t)$; // Train with pseudo-labels L_t as ground truth

Output: Trained model M

Our method for this step is based on DeepCluster[12] whose approach uses a CNN to predict a feature vector for an input image. The CNN is initialized randomly and applied to all images of the dataset. We thus obtain a dataset of feature vectors. Those vectors are then clustered using k-means with a high number of clusters (k=10000 in DeepCluster[12]). Each feature vector is thus assigned a cluster, which now acts as a pseudo-label class. Finally, a cross-entropy loss is computed between those pseudo-labels as ground truth and the predicted feature vectors. The error is back-propagated through the CNN and the weights are adjusted so that the CNN predicts better the pseudo-classes given by the clustering step. This process is repeated, alternating the clustering step and network learning step until convergence.

We used that method with a U-Net[1] which is a fully-convolutional network so that we can compute a feature vector $\boldsymbol{f} \in \mathbb{R}^f$ for each pixel: the samples are now individual pixels instead of whole images but the principle stays the same. However, we hit a scaling problem: for a dataset of 40 images of size 5000 $\times$ 5000 px, we get 1



(a) Original (b) Clusters

Figure 3: Example of clustering.

billion samples and thus 1 billion points in $\mathbb{R}^f$ (with $f = 64$) to cluster. Just to store in memory all those points is prohibitive ($\approx$ 2 terabytes). To cluster them would take a very long time and as this has to be done for every iteration while training, it is unfeasible. Regular gradient descent also suffered from a scale problem and mini-batch gradient descent was invented to get a workable algorithm. We thus use the same mini-batch technique to divide the clustering in manageable chunks: clustering is performed on a batch prior to back-propagation, which is then performed on smaller batches. In order for the clusters to be consistent across batches, we initialize the centroids of the k-means algorithm with the centroids computed from the last batch. See Fig.1 for an overview of the unsupervised feature learning and Alg.1 for more details.

While a simple RGB image can only store the information about the pixels color intensities the idea is that in the feature space every pixel has information from neighboring pixels as well, forming a spatial context. The U-Net architecture was chosen for this task as it is able to capture rich contextual information using a comparatively low number of trainable parameters (less than a million).

2.2 Interactive Segmentation from Sparse Human Annotations

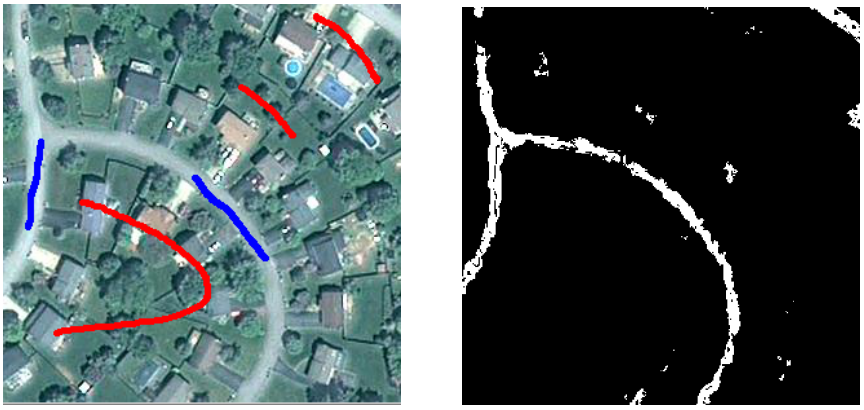
The pseudo-labels are sufficient to train the network and to capture the spatial information around the pixels. However, after applying the trained CNN on the set of unseen images, the output will not give us the desired binary or multi-class segmentation mask. Instead, pixels will have corresponding feature vectors of size f which can be clustered once more with k-means (see Fig.3). Those clusters group semantically similar pixels together but they do not give information about what specific class this pixel belongs to because no class knowledge was introduced in the algorithm thus far.

This class information is added in the second step of our method. It takes as input human strokes for each class of interest and propagates those into a dense segmentation using the features obtained previously (see Fig.4). The goal is to essentially classify each cluster, instead of each pixel like in traditional pixel-wise segmentation. In the case of binary classification, this is done by combining clusters labeled as foreground to a single positive class and those labeled as background to the negative class. The input human strokes give a classification of a few pixels, the vast majority of pixels are negative unclassified. For labeling each cluster, 3 cases arise:

1. The cluster only contains positive and unclassified pixels (with the fraction of positives above a certain threshold t): cluster is positive.

2. The cluster only contains negative and unclassified pixels (with the fraction of negatives above a certain threshold t): cluster is negative.

3. The cluster contains positive, negative and unclassified pixels: cluster is unclassified.



(a) Sparse labels (blue: roads, red: background)

(b) Binary segmentation of roads vs background

Figure 4: Pixel-wise classification using sparse human input strokes and the learned clusters.

For the third case, the cluster in question is split in two, with two new centroids replacing the old one. Those two centroids are initialized at the average position of the positive and negative pixels of the centroids respectively. Then more iterations of k-means are performed to adjust those new centroids. This process of labeling clusters and splitting clusters if necessary is repeated as necessary until the clusters are labeled according to the human inputs. This process is interactive because at any point the human annotator can update their strokes to guide the classification process.

3. EXPERIMENTAL SETUP

3.1 Dataset

The big strength of the unsupervised feature learning step is that it can be used to classify any object type present in a satellite image. In this particular experiment we will segment trees in cities and city suburbs.

As a dataset for our project we used satellite images of 9 cities worldwide: Amostra, Bobo-Dioulasso, Lagos, Pittsburgh, San Diego, Seoul, Zagreb, Tashkent, and New-York (see Fig.5 for two sample crops of the dataset). All images have a ground resolution of 50 cm/pixel and were provided by the industrial company LuxCarta. A part of Pittsburgh was labeled manually for testing while all the other images were used for training. Every city has a different color distribution which makes the task more challenging but closer to real life.

Additionally, each image has an imperfect ground truth for trees. That ground truth was only used to train the fully-supervised model for comparison and was never seen by our method.

(a) New York

(b) Pittsburg

Figure 5: Dataset sample crops.

3.2 Pre-processing

We normalized the image values with mean and standard deviation values for the 3 RGB channels computed on the training dataset which makes the network train better. As satellite images can be very large and of varying sizes, we cut them in patches of constant size 512 × 512 px. In total, 721 patches were used for training 24 testing.

3.3 Pre-Evaluation metric

The Dice Similarity Coefficient (DSC) was used in order to evaluate the overlap between the predicted mask and the manually-labeled ground truth mask. The Dice coefficient is a commonly used evaluation metric for segmentation tasks. It is defined as:

$$dice = \frac{2 * |G \cap P|}{|G| + |P|},$$

where G stands for ground truth and P for prediction. A Dice coefficient of 0 means the predicted mask does not agree with the ground truth at all while a value of 1 means the predicted mask corresponds exactly with the ground truth.

3.4 Hyper-parameters

We set the number of clusters k to 800, clustering batch size b_c to 16, training batch size b_t to 4 and patch size to 512 px. The threshold t for classifying positive and negative clusters is tuned by the user during the interactive labeling of step 2. We found the value $t = 0.3$ to work best for the test image.

4. RESULTS & DISCUSSION

(a) Manual ground truth for trees (b) Trees detected by ClusterNet

Figure 6: Result of ClusterNet on a crop of the test image.

In Fig.6 we can visually observe and compare the output of our ClusterNet approach for tree segmentation in comparison with the ground truth mask labeled manually. We can observe that most ot the trees were located correctly and their shape was preserved. However, the ClusterNet mask is more sparse and contains false positives and negatives. False positive predictions are especially present near the green fence of houses. This might be

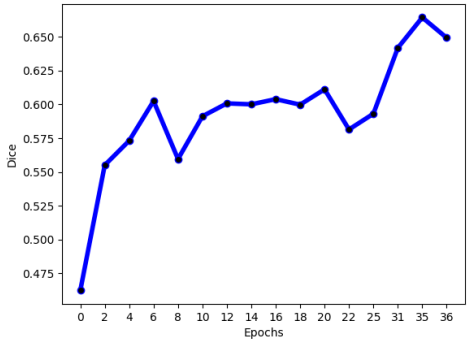


Figure 7: Dice coefficient over training epochs (masks were obtained with the same stroke annotations).

due to the fact that green fences are pretty similar in aspect compared to a tree line. Fig.8 shows the result of our method on the full test image, to be compared with the manual ground truth in Fig.9.

In order to evaluate the progression of the performance of the learned features we used fixed input human-made strokes to perform step 2 of our method after each epoch of training of step 1. This essentially allows us to study step 1 independently of step 2. We can see in Fig.7 the progression of the Dive coefficient during training. The maximum Dice coefficient is achieved after the 35th epoch at 66%. The prediction accuracy (true positive rate) at the same stage is 86%.

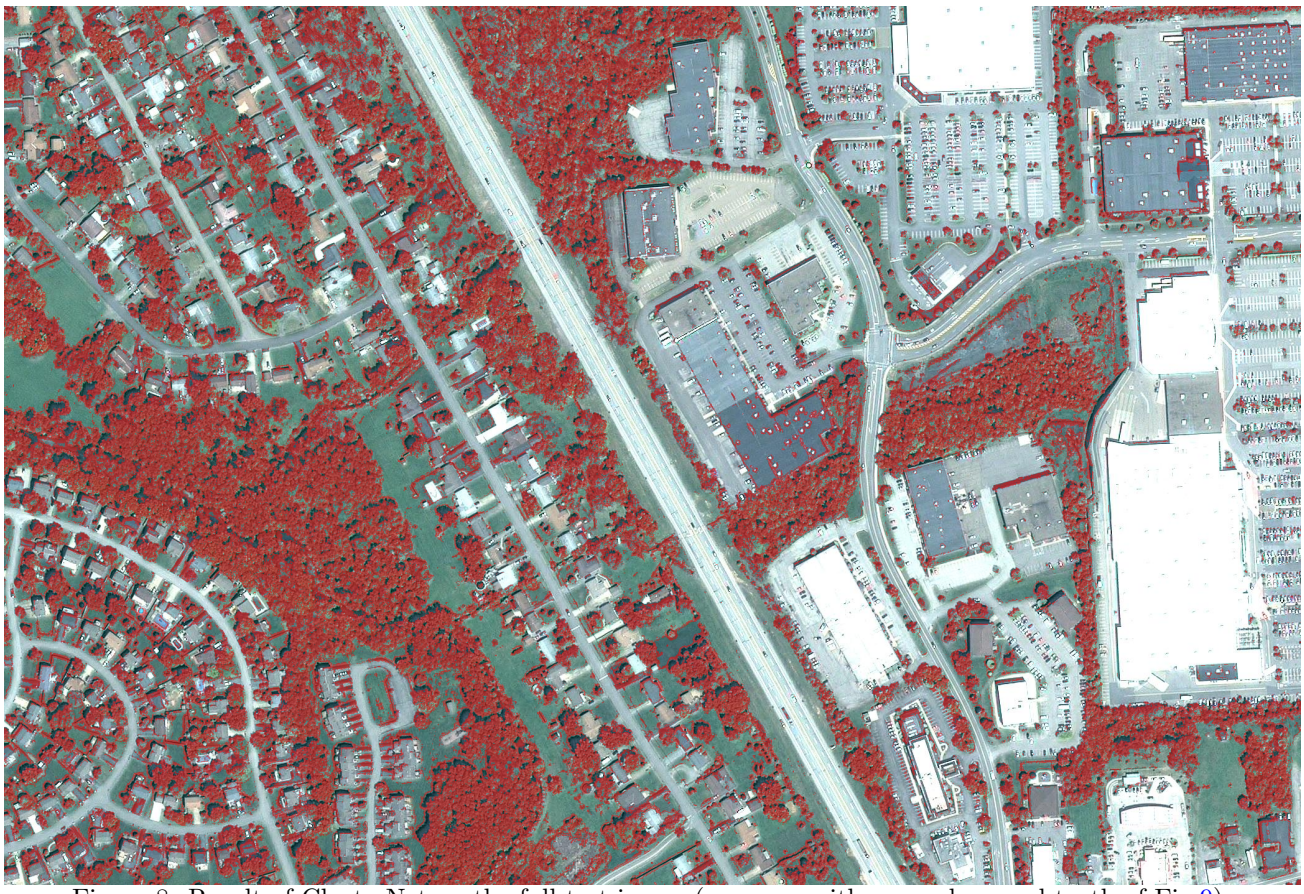


Figure 8: Result of ClusterNet on the full test image (compare with manual ground truth of Fig.9).

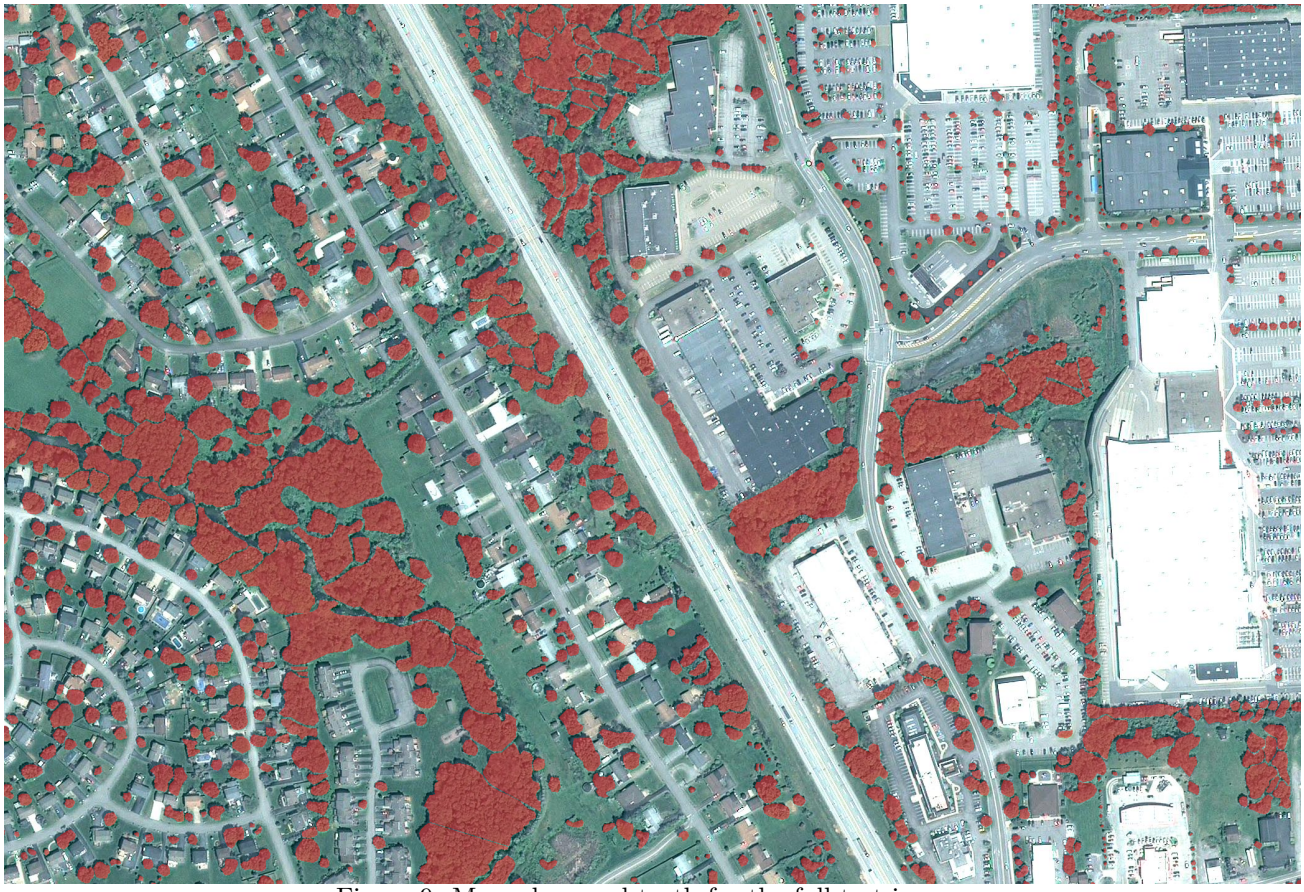


Figure 9: Manual ground truth for the full test image.

We then compare our proposed method with the following ones:

1. Fully Supervised U-Net trained on an imperfect ground truth (such as Fig.2)
2. RGB K-means: clustering applied directly on the RGB test image, followed by our step to for cluster classification
3. Support Vector Machines (SVM)
4. Modified version of Graph Cut (Boykov Graph Cut[18]), using a SVM for computing edge weights[17] (A code on GitHub uses the same method : svm-GraphCut)
5. Superpixel Graph Cut[19]

In Fig.10 we can visually compare between ClusterNet, supervised U-Net, K-means on pixel colors and the ground truth. The prediction by supervised U-Net is less sparse but contains a lot false positives, particularly for grass. This issue is even more remarkable in the case of RGB K-means because it has only color information as input and grass and trees are pretty similar in color. Table 1 shows the results between all methods. As we can see ClusterNet outperforms all other approaches by a significant margin. An interesting fact is that RGB k-means has almost the same performance as our ClusterNet algorithm after 1 epoch of training. See Fig. 7, epoch 1 gives a Dice coefficient of 0.49, compared to 0.48 for RGB k-means. This shows that the pseudo-labels generated in the first epoch of training mostly just accounts for color information of pixels, prioritizing the first skip connection of the U-Net model. Further training epochs help the network to include spatial information around each pixel, allowing ClusterNet to better segment objects of interest.

(a) Imperfect ground truth.

(b) ClusterNet: 66% Dice.

(c) Supervised U-Net: 61% Dice.

(d) K-Means Clustering: 48% Dice.

Figure 10: Visual comparison between the Ground Truth mask, and masks generated by ClusterNet, Supervised U-Net, and K-Means Clustering.

	Dice	Accuracy
SVM	0.18	0.69
SVM + Boykov Graph Cut[18]	0.19	0.71
SuperPixel Graph Cut[19]	0.37	0.68
RGB K-means	0.48	0.76
Supervised U-Net	0.61	0.84
ClusterNet (ours)	**0.66**	**0.86**

Table 1: Comparison table.

5. CONCLUSION

We proposed a method for faster image segmentation by human annotators. Classically an annotator would have to label each pixel almost individually (using strokes). In our ClusterNet method we first general features for each pixel, allowing us to group semantically similar pixels together in clusters. Then the second step leverages human input strokes to classify the clusters. Because annotators now classify clusters instead of pixels, the process in much faster by several orders of magnitude (25 million pixels for just one image vs 800 clusters). Additionally we believe methods with a human in the loop are the most practical in reality as they allow fast corrections by a human expert, guaranteeing a very good final precision.

As future work, it would of course be possible to use more sophisticated clustering techniques like Gaussian Mixture Models (GMMs). It might improve ClusterNet with pseudo-labels of a better quality and might need less number of clusters for the same clustering performance. Less clusters would mean faster cluster classification by annotators later on in step 2 of our method. Another improvement would be to better initialize the network: for now it is randomly initialized which gives relatively random initial clusters. The network then learns to reinforce those clusters and might not be able to recover errors of clustering made early on. A possible improvement would be to use an decoder model after the learned features in order to add a reconstruction loss which would ensure the learned features do not lose information.

The code is available on GitHub: https://github.com/zhygallo/clusternet_segmentation.

6. ACKNOWLEDGMENTS

This work benefited from the support of the project EPITOME ANR-17-CE23-0009 of the French National Research Agency (ANR). We also thank LuxCarta for providing satellite images with corresponding ground truth data.

REFERENCES

[1] Ronneberger, O., Fischer, P., and Brox, T., "U-net: Convolutional networks for biomedical image segmentation," *Medical Image Computing and Computer-Assisted Intervention (MICCAI)* (2015).

[2] Mohney, D., "Satellite imaging: Petabytes of developer & business opportunities," (2018).

[3] Wang, Y. and Yao, Q., "Few-shot learning: A survey," (2019).

[4] Khoreva, A., Benenson, R., Hosang, J., Hein, M., and Schiele, B., "Simple does it: Weakly supervised instance and semantic segmentation," *Conference on Computer Vision and Pattern Recognition (CVPR)* (2017).

[5] Remez, T., Huang, J., and Brown, M., "Learning to segment via cut-and-paste," *The European Conference on Computer Vision (ECCV)* (2018).

[6] Lin, D., Dai, J., Jia, J., He, K., and Sun, J., "Scribblesup: Scribble-supervised convolutional networks for semantic segmentation," *Conference on Computer Vision and Pattern Recognition (CVPR)* (2016).

[7] Bearman, A., Russakovsky, O., Ferrari, V., and Fei-Fei, L., "What's the Point: Semantic Segmentation with Point Supervision," *The European Conference on Computer Vision (ECCV)* (2016).

[8] Rakelly, K., Shelhamer, E., Darrell, T., Efros, A. A., and Levine, S., "Few-shot segmentation propagation with guided networks," *CoRR* **abs/1806.07373** (2018).

[9] Zhang, W., Zeng, S., Wang, D., and Xue Shanghai, X., "Weakly supervised semantic segmentation for social images," (2015).

[10] Pathak, D., Krhenbhl, P., and Darrell, T., "Constrained convolutional neural networks for weakly supervised segmentation," *International Conference on Computer Vision (ICCV)* (2015).

[11] Durand, T., Mordan, T., Thome, N., and Cord, M., "WILDCAT: Weakly Supervised Learning of Deep ConvNets for Image Classification, Pointwise Localization and Segmentation," *Conference on Computer Vision and Pattern Recognition (CVPR)* (2017).

[12] Caron, M., Bojanowski, P., Joulin, A., and Douze, M., "Deep clustering for unsupervised learning of visual features," *The European Conference on Computer Vision (ECCV)* (2018).

[13] Moriya, T., Roth, H. R., Nakamura, S., Oda, H., Nagara, K., Oda, M., and Mori, K., "Unsupervised segmentation of 3d medical images based on clustering and deep representation learning," *CoRR* **abs/1804.03830** (2018).

[14] Yang, J., Parikh, D., and Batra, D., "Joint unsupervised learning of deep representations and image clusters," (2016).

[15] Xia, X. and Kulis, B., "W-net: A deep model for fully unsupervised image segmentation," *CoRR* **abs/1711.08506** (2017).

[16] Rother, C., Kolmogorov, V., and Blake, A., "Grabcut -interactive foreground extraction using iterated graph cuts," *ACM Transactions on Graphics (SIGGRAPH)* (2004).

[17] Tarabalka, Y. and Rana, A., "Graph-cut-based model for spectral-spatial classification of hyperspectral images," *International Geoscience and Remote Sensing Symposium (IGARSS)* (2014).

[18] Boykov, Y. and Funka-Lea, G., "Graph cuts and efficient n-d image segmentation," *International Journal of Computer Vision* (2006).

[19] Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and Ssstrunk, S., "Slic superpixels," (2010).