Group 97: Road Segmentation

Laurent Lejeune, Tatiana Fountoukidou, Guillaume de Montauzon

I. Introduction

In this project we were given a set of 100 RGB satellite images, and a corresponding binary mask for each one of them, indicating the regions of the image that represents road. Our task was, given an unknown satellite image to perform automatic segmentation of the roads, marking everything else as background. We therefore have to solve a binary classification problem, effectively for every image pixel. An example of an image and its ground truth is shown in figure 1.

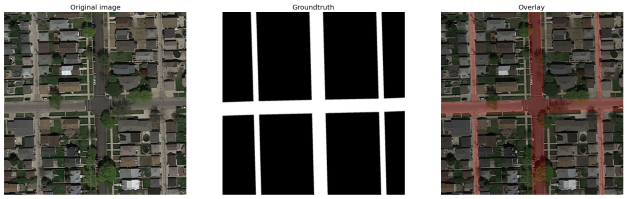

Figure 1: Example of training image with its ground-truth

II. Baselines

To begin with, we set some simple baselines for tackling the problem of road segmentation. The baseline selection is described below.

A. Baseline classifier selection

As a baseline, every image was broken down to a number of non-overlapping 16×16 image patches. If more than 25% of the groundtruth corresponding to the patch was road, the patch was labeled as road. This way, we ended up with a single label for every patch, thus simplifying the classification problem. As a feature, a vector of size 2 was used, containing the mean value and the variance of the pixels contained in the patch (A feature vector of size 6, consisting of the mean and the variance for each color channel was also tried, but it lead to no improvement). A grid search to indicate a simple classifier that best works for this setup was run, and the one with the best performance was set as a baseline for further improvement. The performance is measured in terms of the F-score, that considers both the precision and recall, and is defined as:

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = 2 \cdot \frac{TP}{2TP + FN + FP}$$

A 5-fold cross validation is run over the training dataset, and the F-score on the testing set is calculated for each fold. The mean and the variance of the F-score over the 5 folds is then calculated. The results of the baseline search are shown in table **??** (only the best parameters of each classifier are demonstrated, for clarity reasons). For the rest of the work, the performance of the random forest (RF) classifier will be used as the baseline to further investigate how these results can be improved [1].

Table I: Baselines

Classifier	$F_1 - score$(**mean** $\pm$ **std**)
Logistic regression ($\lambda = 1e5$)	0.44 ± 0.02
SVM (rbf kernel, $\lambda = 1e5$)	0.5 ± 0.02
RF (number of trees = 50, max depth = 10)	$\mathbf{0.5 \pm 0.02}$

B. Patch size selection

We examined the impact of the patch size on the classification performance. The mean and the variance of the pixels was again used as a feature, and the classifier was the baseline RF described above. The F-score was recorded for a 5-fold cross validation, and the results for the different patch sizes are shown in table II. Even though the larger the patch size, the better the

Patch Size	**F-score (mean $\pm$ std)**
8	0.45 ± 0.03
16	0.5 ± 0.03
25	0.54 ± 0.04
32	0.55 ± 0.06

Table II: Patch size comparison

[1] Although the RF and SVM classifiers gave similar F1 scores, the RF was significantly faster to train. We therefore chose to use it for the remaining of our experiments.

F-score, with a quick inspection of the data we realized that many roads were too narrow to be represented by a larger patch size. The higher score in this case is misleading, since it is calculated on a patch level, and not on a pixel level. This way, although more patches are classified correctly, the overall segmentation mask that comes up is not representative enough. For this reason, the patch size of 16 was kept, as more appropriate to capture the different road widths.

III. Data exploration

The provided training set contains 100 images of size 400x400 along with their ground-truth. A total of 6 images are discarded because they either show a too small quantity of positive class pixels, or some misleading regions such as rail-tracks. We notice that most images are made of grid-like roads, sometimes occluded by trees.

IV. Mid-level segmentations

The image pixels are first grouped in two different manners:

1) Square patches: The image is divided in non-overlapping patches of size 16x16.
2) SLIC Superpixels (Simple Linear Iterative Clustering) [1]: Pixels are grouped in mid-level regions in an iterative manner. The algorithm starts from a regular grid of cluster centers and iteratively updates the labels of their neighboring centers based on a distance measure. This method improves over the square patches method because the pixels are already pre-segmented. Their feature vector will therefore be easier to discriminate.

V. Feature extraction

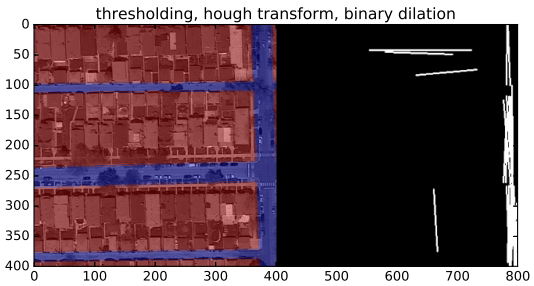

Figure 2: Example of a hough transform. Left: Input image with the ground-truth overlay. Right: 20 lines with lowest color variance.

Following an exploration of the related litterature, we select a set of features to extract.

- SIFT (Scale-Invariant Feature Transform) [2]: This descriptor is used extensively in computer-vision applications. It computes a histogram of oriented gradients on 16x16 windows centered at a keypoint and gives a descriptor of 128 scalar values. The keypoint detection step is not performed, instead we extract the descriptors on a dense grid at canonical scale and orientation. As advised in [2] to improve illumination invariance, the integer value descriptors are first normalized to unit-norm, ceiled to 0.2, and renormalized to unit-norm. As we require that each segment be represented by a single feature vector, we encode the dense SIFT descriptors contained in a given segment in a "bag-of-features" manner through the following steps:
  1) Based on a sufficiently large number of SIFT descriptors computed on 10 images, we start by fitting a PCA model. We have checked that the explained variance at 60 components is above 99%.
  2) A codebook is generated on the aforementioned training samples. A codebook is merely a set of K-means clusters that is used to encode the input (compressed) descriptors to integer values.
  3) We then compute a normalized histogram of codes (bag-of-features) in each segment. This gives us a single texture feature vector for mid-level regions.
- Hough line transform: This transformation has already been used in a state-of-the-art method [3]. First, the edge map is computed using a canny edge detector. Given some parameters, a set of lines are extracted on the edge maps and sorted based on their RGB variance, i.e. we want to keep the lines along which the color variations is minimal. An example is shown on figure 2. As feature, we take the mean value of the hough map on the segment.
- Euclidean distance transform. This straightforward transform is used to compute, at each pixel location, the shortest "taxicab" distance to an edge pixel. Using as input the canny edge map, we expect this feature to discriminate cluttered regions such as those containing blocks of buildings where the Euclidean distance tends to be smaller. The mean value over the segment is used.
- Mean RGB value. Roads tend to have greyish colors.

To summarize, our feature extraction procedure provides a total of 65 features per superpixel.

[2]https://people.csail.mit.edu/hasinoff/320/sift-notes.txt

A. Refinement of generic models using structured SVM

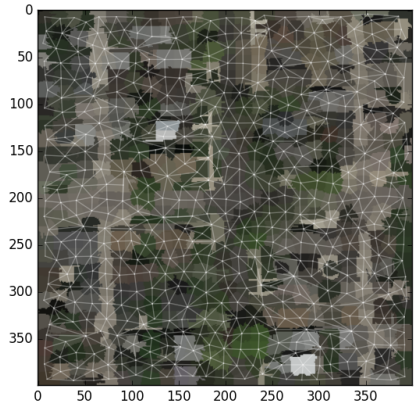

Figure 3: Example of a superpixel-segmented image with connecting edges.

Using structured models [4], one can leverage the spatial relations between mid-level regions. As shown on figure 3,a segment considered as road gives a strong prior on the "roadness" of its neighboring segment. This is formalized as an undirected graph on which the node features are assigned unary potentials. In our case, the unary potentials are given by probability estimates given by generic models such as logistic regression or random forest. Inspired by [5], the edge costs are made off of two features: The difference in mean LUV color, and the number of pixels that separate two segments (length of separating path). This last feature allows to penalize segments that are "weakly" connected. Indeed, we have verified visually that roads tend to be composed of regular chains of square-like segments, thereby justifying that choice.

Formally, structured models aim at maximizing an energy functions of the form:

$$\begin{aligned} E_w(X,Y) &= \sum_{i\in\mathcal{V}} E_{data}(y_i;x_i) + \sum_{i,j\in\mathcal{E}} E_{smooth}(y_i;y_j) \\ &= \mathbf{w}^T\psi(X,Y) \end{aligned} \tag{1}$$

Where $\mathcal{V}$ is the set of vertices representing a segment, $\mathcal{E}$ are the edges. The data and smoothness term are combined in the joint-features vector ψ. The variable y represents the structured labels. In our setup, any probabilistic regression model (logistic regression, random forest,...) can be used for the data term. Following [5], the pair-wise edges potentials are given by:

$$\phi(c_i,c_j|s_i,s_j) = \frac{L(s_i,s_j)}{1+\|s_i-s_j\|} \tag{2}$$

Where c and s are the mean LUV-space colors. The function L expresses the length of the shared boundaries between two segments. The provided code relies on the pyStruct package [6], which implements the cutting-plane algorithm proposed by Tsochantaridis et al [4].

VI. Classification

2 approaches were followed for this project. the first consists of feature extraction and classification, with the features mentioned aboved. The second consists of training a convolutional neural network (CNN) that takes as input the image and outputs a segmentation mask. Both approaches will be described below.

A. Classification based on feature vectors

The features described above were used to train a random forest ensemble classifier. A grid search to define the hyperparameters of the classifier was performed. The chosen parameters were the following:

$$\text{Number of trees} = 50$$

$$\text{Maximum tree depth} = 10$$

The CRF refinement step was then performed.

B. CNNs

1) Guillaume for you!!!:

2) U-net: A convolutional neural network designed for image segmentation was used, as described in [7]. U-net is a fully convolutional neural network, meaning that it consists exclusively on convolutional and pooling layers, and not fully connected layers in the end. As a result, the output of the network is not a class assignment for the input, but a feature map. In our case the output of the network is a segmentation mask for the input image. The structure of the network is shown in figure 4.

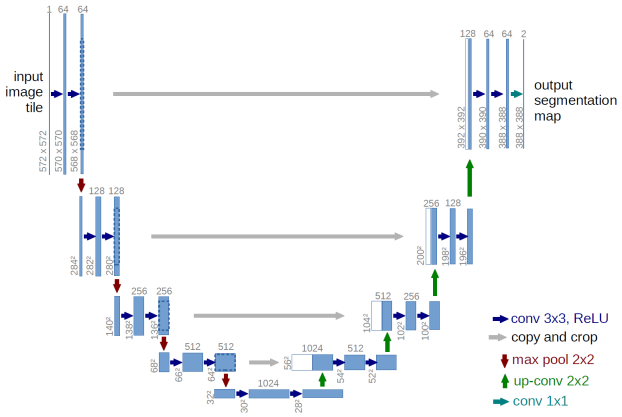

Figure 4: U-net [7]

Classification method	$F_1 - score$(**mean** $\pm$ **std**)
on mean and variance features [Baseline]	0.5 ± 0.02
Logistic regression with $\lambda = 1e5$ on feature vectors	0.64 ± 0.02
RF on feature vectors without refinement	0.68 ± 0.03
RF with CRF refinement	0.79 ± 0.02
U-net	

Table III: Concentrated results

VII. Results

The results of a 5-fold cross validation, for the different classification approaches that were attempted and described above are shown in table III

VIII. Conclusions

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. 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.

[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[3] Y. Lv, G. Wang, and X. Hu, "Machine Learning Based Road Detection from High Resolution Imagery," *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pp. 891–898, June 2016.

[4] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables," *Journal of Machine Learning Research*, vol. 6, no. Sep, pp. 1453–1484, 2005.

[5] B. Fulkerson, A. Vedaldi, and S. Soatto, "Class segmentation and object localization with superpixel neighborhoods," in *2009 IEEE 12th International Conference on Computer Vision*, pp. 670–677, Sept 2009.

[6] A. C. Müller and S. Behnke, "Pystruct: learning structured prediction in python.," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 2055–2060, 2014.

[7] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *arXiv preprint arXiv:1505.04597*, 2015.