CIL Road Segmentation Project Report

Lixin Xue, Yu Fei, Hongjie Chen, Shengze Jin
Group: Veggie Chicken
Department of Computer Science, ETH Zurich, Switzerland

***Abstract*—Training a promising model with limited data for a specific task like road segmentation is very challenging for recent deep learning models. In this paper, we tackle this problem from the view of base model and the view of post-processing. To provide generalizable predictions, we trained our base model with several data augmentation techniques and two auxiliary tasks, road edge prediction and road center line prediction, through which the model learns better representation. The outputs of the base model is then processed by a Conditional Random Field (CRF) with a novel direction-based kernel which exploits the geometry of roads. The CRF is supported by an extra probability propagation module to relieve undersegmentation. Our approach achieved 92.40% and the 3rd place on the public leaderboard.**

I. Introduction

Image segmentation is a fundamental task in computer vision. Road segmentation is a special case of it and has its own properties. Recently, deep learning models demonstrate impressive performance in the segmentation tasks while requiring a large number of training samples. However, in our setting, the data is very limited, and deep models is prone to overfitting. To overcome this, we come up with several modifications in the base model and a post-processing module consisting of several novelties.

For the base model, we use data augmentation and multitask learning to alleviate the problem of overfitting. Specifically, we train our network to predict road edges and road centerlines in addition to the original road surface prediction following the idea in RoadNet [1]. The road edges and centerlines can be easily extracted from ground truth segmentation maps with existing methods.

The other piece of our approach is a post-processing module that utilizes the geometry of roads. Specifically, we exploit the power of CRF, which is still playing an important role in segmentation tasks despite the great success of the deep models. CRF benefits segmentation tasks with its potential to integrate prior knowledge that captures relationships between object classes and, as our method explored, the geometry of specific objects. However, one limitation of many existing kernels used for CRF (like in [2]) is that they are designed for general objects based on a very rough idea that points that are similar (i.e. close in distance and similar in color) to each other are likely to be in the same class. The directional homogeneity of this setting scarifies the abilities to capture more complex information for objects like roads in a more specified task.

Intuitively, roads are objects that mainly cover stripe-like regions, which usually extend long alone one direction while being quite narrow in the perpendicular direction. This motivates an extension of "similarity": points on the same road are similar in the sense that they lie roughly along the same line. We argue that a reasonable way to model "lying along the same line" mathematically is the following: first assign each pixel a direction which is likely for the pixel to be in if it is on a road; then compare the "distance" between the lines determined by pixels and the corresponding directions. Based on this similarity measurement, we developed a novel direction-based kernel for CRF that is able to capture the geometry of roads. We also proposed an easy but efficient way to generate the direction map.

Another challenge faced by CRF is that he performance of CRF largely depends on the probability map provided by the base model, while the deep base models suffer from limited data. The dependence is more severe when the number of pixels in different classes are imbalanced. In an image, road pixels are usually much fewer than the background ones. Since the probability of a pixel will be strongly connected with pixels "around" after CRF, the class imbalance can cause problems. This effect results in that normally there will be fewer road pixels in the refined probability map, which may rule out narrow roads captured by the base model with weak probabilities.

Although, we can compensate for this by playing with the parameters of CRF (mainly the compatibility matrix proposed in [2]), extreme parameters may generalize poorly. Motivated by the observation that CRF does good a job in shrinking and refining the potential road regions, we developed an extra probability propagation module right after the base model to give CRF an input that is easier to identify and complement roads.

The main contributions of this work can be summarized as below:

1) We extend the PSPNet model with auxiliary edge and centerline prediction tasks as done in RoadNet [1].
2) We introduce a probability propagation module to refine the probability map before passing it to CRF.
3) We propose a direction-based kernel specific for our road segmentation tasks.
4) We achieve a score of 0.9240 on the public test data, ranking 3rd in the leaderboard.

II. Base Model

A. Preprocessing

Having a large dataset is crucial for the performance of deep learning models. However, in this task we are provided with 100 images only. Therefore, we need to use data augmentation to create more training samples. Specifically, we randomly rotate and mirror the training images so that we have images of roads in all directions. Since there are different types of roads with different scales, we also apply random scaling together with random crop and padding. Experiment results demonstrate the effectiveness of our data augmentation for the prediction accuracy.

B. PSPNet

We use PSPNet [3] as our base segmentation model, which is the winner in ILSVRC Scene Parsing Challenge 2016 [4]. For the backbone CNN, ResNet is used with dilated convolution as introduced in DeepLab [5] to extract feature maps. Then, features are pooled at different scales and then 1×1 convolution is performed for each pooled feature map to reduce the number of channels. Next, the feature maps are bilinearly upsampled and concatenated with the original feature maps to aggregate the overall context. Finally, the fused feature maps are passed to a decoder to predict the segmentation map. Another important detail is that auxiliary loss is used in training to supervise the middle level feature maps, which is similar to the auxiliary classifier in GoogLeNet [6].

C. PSPNet2

When trained on one single task, a model might ignore information that helps it do even better on the target metric. Specifically, this information comes from the training signals of related tasks. By sharing representations between related tasks, we can enable our model to generalize better on our original task. Therefore, we apply the idea form RoadNet [1] to add two auxiliary tasks: road edge prediction and road centerline preidction. Specifically, with feature maps from PSPNet and original images as input we use another two CNNs to predict the segmentation map for road edges and road centerlines. The groundtruth for road edges are extracted by a Canny edge detector [7] from the original road groundtruth. The centerlines are also extracted from the road groundtruth using Zhang-Suen thining algorithm [8]. With the groundtruth for all three tasks (the original road surface prediction, road edge prediction, and road centerline prediction) available, we train the parameters of three CNNs in an end-to-end manner, where we use balanced cross entropy loss for the edge and centerline prediction tasks.

III. Post-processing

One of the main contributions of our work is a novel direction-based kernel for CRF supported by an extra probability propagation module. In this section, we will briefly introduce CRF and discuss each component of our method.

A. CRF

Briefly speaking, a fully connected CRF for image segmentation aims at minimizing an energy function over the labeling $\mathbf{X}$ defined by a combination of unary and pairwise terms (for a more detailed introduction refer to [2]):

$$E(\mathbf{x}) = \sum_i \psi_u(x_i) + \sum_{i<j} \psi_p(x_i, x_j). \tag{1}$$

The unary potential $\psi_u(x_i)$ is the inverse likelihood of assigning label x_i to pixel i and can be obtained from our base CNN model. We model the pairwise potentials by

$$\psi_p(x_i, x_j) = \mu(x_i, x_j) \sum_{m=1}^{K} w^{(m)} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j), \tag{2}$$

where $k^{(m)}(\mathbf{f}_i, \mathbf{f}_j) = \exp\left(-\frac{1}{2}(\mathbf{f}_i - \mathbf{f}_j)^T \Lambda^{(m)} (\mathbf{f}_i - \mathbf{f}_j)\right)$ is a Gaussian kernel with a diagonal covariance parameter matrix Λ; $\mathbf{f}_i$ is a feature vector; $w^{(m)}$ is the weight for kernel m; μ is a label compatibility function that allows the model to treat different label pairs differently. We choose the number of kernels $K = 3$ for this road segmentation task. Specifically, we have the first and the second kernel the same as the the appearance kernel and the smoothness kernel in [2]:

$$k^{(1)}(i, j) = \exp\left(-\frac{\|\mathbf{p}_i - \mathbf{p}_j\|^2}{2\sigma_\alpha^2} - \frac{\|\mathbf{I}_i - \mathbf{I}_j\|^2}{2\sigma_\beta^2}\right), \tag{3}$$

$$k^{(2)}(i, j) = \exp\left(-\frac{\|\mathbf{p}_i - \mathbf{p}_j\|^2}{2\sigma_\gamma^2}\right), \tag{4}$$

where $\mathbf{p}_i = (x_i, y_i)$ is the position of pixel i and $\mathbf{I}_i$ is the corresponding rgb color vector. The third kernel exploits the geometry of roads and will be discussed in more details in the following sections. The optimization of CRF can be done efficiently with method proposed in [2], which is compatible with our novel structural kernel as well.

B. Direction-based Kernel

Suppose we already have the direction map that assigns a direction d_i to each pixel i in a image. The direction d_i together with the position $\mathbf{p}_i$ induce a line l_i for pixel i. Let x_i, y_i be the intersections of line l_i with the x and y axis. Note that we can set $y_i = \infty$ ($x_i = \infty$) for a vertical (horizontal) line. Motivated by the observation that "close" lines should have close intersections, we can measure the distance between two lines l_i and l_j by

$$\text{dist}(l_i, l_j) = |y_i - y_j|^2 + |x_i - x_j|^2. \tag{5}$$

With this distance measurement of lines, we can define a kernel that captures the stripe-like geometry of roads by

$$k_s(i, j) = \exp\left(-\frac{\text{dist}(l_i, l_j)}{2\sigma_s^2} - \frac{\|\mathbf{p}_i - \mathbf{p}_j\|^2}{2\sigma_p^2}\right), \tag{6}$$

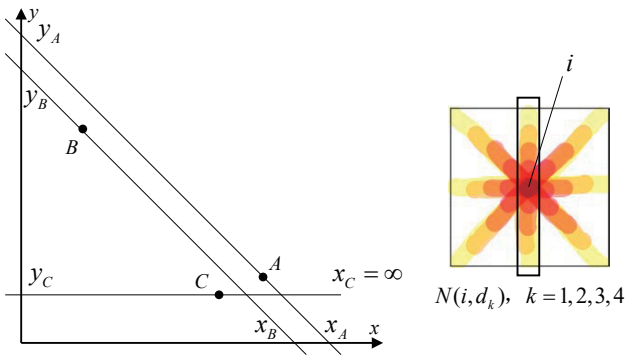

Figure 1. Left: Point A, B and C with their directions determine three lines, where l_A is more "similar" to l_B than l_C. Despite the stronger closeness of point A and C in distance, A and B will be more similar in terms of our direction-based kernel defined in (6), which agrees with the fact that they are potentially more likely to be on the same road. Right: The direction of pixel i is determined by the sum of similarity defined by (3) between i and j's in its neighborhood along 4 typical directions.

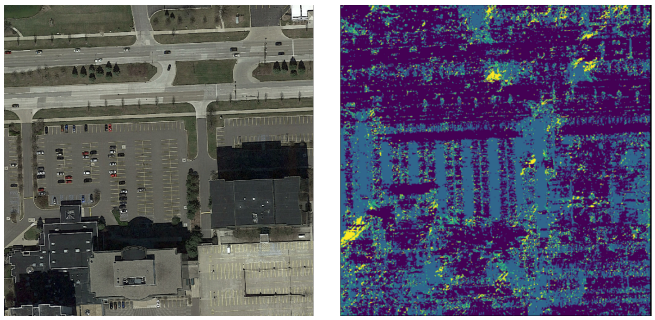

Figure 2. Left: original image. Right: extracted direction map with different colors correspond to different directions.

where the second term penalizes the contribution from points that are too far. This kernel gives strong similarity to points that are potentially around the same line and thus likely to be on the same road. The way this kernel works is illustrated in the left part of Figure 1. Note that our proposed kernel takes an analogous form with the kernels in [2] (intersections (x_i, y_i) as color vectors (r_i, b_i, g_i)), therefore it can be integrated into the optimization framework proposed in [2].

To simply the problem, we utilize an important observation that all roads in both the training and the test set are approximately along one of the horizontal, vertical, diagonal, and anti-diagonal direction. Therefore we can choose d_i to be one of these four typical directions.

C. Direction Extraction

All the discussion above is based on an assumption that we can somehow obtain a pixel-wise potential direction map for each image. With the observation of typical directions, we show that we can do this in a very simple way which gives a quite good result as illustrated in Figure 2.

The main idea is that the direction along which pixels near i are most similar to i should be a good choice for d_i

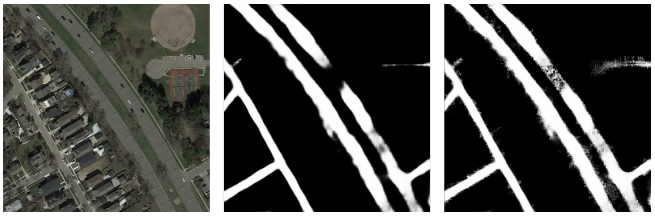

Figure 3. Left: original image. Middle: base model probability. Right: iteratively propagated probability along horizontal and vertical directions.

since intuitively pixels on the same road have roughly the same color. Specifically, we compute for each pixel i in the original image the sum of similarity between i and pixels nearby along each typical direction via the measurement induced by the appearance kernel (3), which takes both the distance and the color into account. The neighborhood, i.e. the nearby pixels, of pixel i along typical directions $N(i, d_k),\ k = 1, 2, 3, 4$ are illustrated in the right part of Figure 1. Mathematically, the direction of i is given by

$$\arg\max_{\substack{d_k \\ k=1,2,3,4}} \sum_{j \in N(i,d_k)} \exp\left(-\frac{\|\mathbf{p}_i - \mathbf{p}_j\|^2}{2\sigma_1^2} - \frac{\|\mathbf{I}_i - \mathbf{I}_j\|^2}{2\sigma_2^2}\right). \tag{7}$$

The size of the neighborhood and the parameters σ_1, σ_2 are chosen empirically. The directions are extracted based on Gaussian blurred images to give a less noisy mapping.

D. Probability Propagation

The idea is quite simple: propagate the probability of each pixel to its neighborhood $N(i)$ in an iterative manner. Specifically, after each iteration

$$P_i' = \max_{j \in N(i)} \{P_j \cdot k_{ij}\}, \tag{8}$$

where k_{ij} is a measure of similarity, and we choose it to have the form of (3) but with different σ's. To make use of the road geometry and accelerate the computation, again we utilize the typical directions observation by further restricting the propagation to only the horizontal and vertical direction (Figure 3). Note that by propagating iteratively the diagonal and anti-diagonal direction are taken care of as well. The map after propagation can be noisy, but remember that CRF is an expert in dealing with noisy inputs.

E. CRF-RNN

Since manually tuning a large number of parameters in CRF is too tedious, Zheng et al. [9] showed that the mean-field inference of CRF [2] can be reformulated as a Recurrent Neural Network (RNN). Thus, instead of using CRF as an offline post-processing method, we can integrate CRF with PSPNet and obtain an end-to-end trainable deep network. Such network is called CRF-RNN and is trained using backpropogation. Importantly, our direction-based kernel is compatible with this framework as well.

IV. Results

Implementation Details. We randomly split the training data into a training set (90 images) and a validation set (10 images). We train on the training set and choose the best checkpoints based on patch prediction accuracy on the validation set. For our base model, we train 2800 epochs using SGD with the learning rate set as 10^{-3}. The parameters of post-processing modules are chosen via random search based on the same accuracy measure on the validation set.

Runtime and Memory Usage: We train our base model with batch size 4, and one epoch on all 100 images takes 12 seconds for PSPNet, 15 seconds for PSPNet2, and 84 seconds for CRF-RNN (the CRF part is currently only supported to run on CPUs). With ResNet50 as backbone, PSPNet and CRF-RNN takes up 5529 MiB GPU memory while PSPNet2 occupies 6841 MiB. For direction extraction, it takes 2 minutes per image with our vanilla Python code. As for post-processing, it takes 10 seconds per image to perform CRF with propagation and the direction-based kernel.

Final Results: Our approach reached accuracy of 0.9240 on the public test set, ranked 3rd in the leaderboard. Our results is significantly better than the provided linear model (0.4092) and naive CNN (0.6158). We run a number of ablation studies to analyze our model. As in the method section, we will discuss them in 2 parts.

Base Model: We train our base PSPNet model with and without data augmentation (DA in the table) to demonstrate the effectiveness of our preprocessing module. We also train PSPNet and PSPNet2 with same hyperparamters and we find PSPNet2 gives slightly better results. When integrated with our post processing module(CRF in the table), PSPNet2 gives us significant better results.

Method	Public Score
PSPNet without DA	0.8837
PSPNet with DA	0.9042
PSPNet2 with DA	0.9065
PSPNet with DA & CRF	0.9194
PSPNet2 with DA & CRF	**0.9240**

Table I
Ablation study results for base models.

Post-processing: Since the post-processing part of our method combines various mechanisms, we have done two experiments shown in Table II in order to provide additional insight into what makes our approach performant. We compared the performance of different combinations of modules, namely the CRF: 2 kernels from [2], propagation module and the direction-based kernel. In both experiment, we use the same base model trained on the training set (90 images), and all post-processing parameters are tuned via random search.

In the first experiment, we tune parameters in the different setting on the validation set (10 image), and we evaluate the performance via the public score. We found that our post-processing model leads to a leap in the score. However, the difference among the settings are not dramatic, and the improvement of the propagation module is not as expected. By analyzing the visualization results we found that our randomly generated validation set happens to include images with parking lot while all of them are not labeled. As a result, all settings containing CRF mainly do a job as suppressing the parking lot in the prediction. This reveals one of the main obstacles of this project: the labeling of the parking lot is not consistent (parking lot in 7 out of 17 training image are labeled), and there are 40 samples containing a large area of parking lot in the test set.

Figure 4. Left: original image. Middle: results using only CRF in [2]. Right: result of our approach. (Parameters of post-processing modules in the middle and the right are both tuned on local validation set)

CRF	Propagation	Kernel	Public Score	Local Test Score
-	-	-	0.9042	0.8809
√			0.9172	0.8882
√	√		0.9098	0.8875
√		√	0.9180	0.8909
√	√	√	**0.9194**	**0.8927**

Table II
Ablation study results for post-processing modules.

To compensate for this, we conduct a second experiment, where we randomly choose 17 test images as a local validation set and 19 test images as a local test set with both of them labeled by ourselves in a consistent manner. We label all parking zones, and the results in Table II shows that in this case our methods show greater advantage over CRF in [2]. As shown in Figure 4, our method can complement roads and producing a finer results as expected.

CRF-RNN: Although it is expected that better performance can be achieved using end-to-end training, the best result CRF-RNN obtained is only 0.8982 on the public test set and 0.9046 on the local validation set. One reason could be that, the size of our dataset remains still small, while CRF-RNN introduces more trainable parameters. As a contrast, more than 10k images are used in [9].

V. Summary

In this work, we try to tackle the problem of road segmentation with limited data in the following two ways: first we train our segmentation network with data augmentation and two auxiliary tasks, i.e. edge and centerline predictions; then we propose probability propagation and a novel direction-based kernel for CRF post-processing, which utilizes the geometry of roads. With these methods, we ranked the 3rd place on the public leaderboard.

REFERENCES

[1] Y. Liu, J. Yao, X. Lu, M. Xia, X. Wang, and Y. Liu, "Roadnet: Learning to comprehensively analyze road networks in complex urban scenes from high-resolution remotely sensed images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 4, pp. 2043–2056, 2019.

[2] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected crfs with gaussian edge potentials," in *Advances in neural information processing systems*, 2011, pp. 109–117.

[3] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *CVPR*, 2017.

[4] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[5] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, 2018. [Online]. Available: https://doi.org/10.1109/TPAMI.2017.2699184

[6] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Computer Vision and Pattern Recognition (CVPR)*, 2015. [Online]. Available: http://arxiv.org/abs/1409.4842

[7] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, no. 6, p. 679–698, Jun. 1986. [Online]. Available: https://doi.org/10.1109/TPAMI.1986.4767851

[8] T. Y. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," *Commun. ACM*, vol. 27, no. 3, p. 236–239, Mar. 1984. [Online]. Available: https://doi.org/10.1145/357994.358023

[9] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr, "Conditional random fields as recurrent neural networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1529–1537.