EPFL Machine Learning Road Segmentation Challenge—Beyond Dream

Yuxuan Long, Yiting Zhang, Haojun Zhu
École Polytechnique Fédérale de Lausanne, Switzerland

***Abstract*—Road segmentation on satellite images is always a challenging task. One challenge could be that the extraction of information from multiple level of scale is needed in order to correctly infer the semantic meaning of the pixels. To deal with it, dilated convolutions are introduced so we choose our work to be based on D-LinkNet [1]. After making some modifications to D-LinkNet, we propose a new network architecture called D-LinkNet+, which outperforms D-LinkNet in the experiment. By applying some data augmentation and test time augmentation, D-LinkNet+ has reached at a F1 score of 0.922.**

I. Introduction

Image segmentation is a popular computer vision problem that has been researched over several decades. It requires us to classify the pixels by their semantic meanings. In this project, we target at a particular scenario— the satellite images, for which the aim is to extract the roads out of the images. In this case, the pixels are either labelled as road (1) or non-road (0). This is equivalent to binary classification, but there indeed exist many challenges. In this project, we are only provided with 100 training images with size 400×400, which is of quite limited size. Second, some roads in the images are hard to visually distinguish even from a human perspective, like the region with dark shadow.

Recently, deep neural network has been a prevailing tool in many research areas, particularly it outperforms the classical methods in many visual tasks. Deep neural network has the advantages that it can extract complex features and semantic information from the images. In this project, we use deep neural network to solve the road segmentation task.

II. Network Architecture

A. Related work

The most common architecture of deep neural network on image segmentation is usually an analogy to auto-encoder, which contains an encoder followed by decoder. The main difference from auto-encoder is that, the output of the decoder is expected to be a multi-labelled mask, instead of a reconstructed signal. In our case, the encoder should transform the original signal into latent feature space that has some semantic information, and the decoder upsamples them back to the original space with spatial details. Since the latent features may lose the spatial information, in modern network architecture, there are usually some shortcut connections between encoder and decoder. U-Net [2] is one popular example, particular known for its great expression power on extremely small dataset.

B. D-LinkNet

In this project, we utilize a more recent work called D-LinkNet [1], which is the winner of CVPR DeepGlobe 2018 Road Extraction Challenge. The architecture of D-LinkNet is shown in figure 1, where the section A is the encoder, section B the dilated convolution layers and section C the decoder. The encoder is just ResNet34 [3] without the final fully connected layer, and the decoder is consisted of a sequence of convolution and deconvolution layers. Note that D-LinkNet is originated from Linknet [4], where the main difference is that D-LinkNet adds dilated convolution layers in the center part (i.e. section B). In figure 1, the center part of D-LinkNet consists of successive dilated convolutions with dilation rate 1, 2, 4 and 8 respectively. Specifically, as proposed in [1], the output of this center part is the sum of the outputs from all dilated convolution, including the features from the encoder.

The dilated convolution layer is the key to win the road segmentation challenge. In some regions of satellite images, pixel classification needs information at multiple level of scales. A hard case can be the walking paths (not roads actually) or the railways spanned in a large region, at this case we may have to search for the global context in the image. Dilated convolution is then deployed to solve those issues, since it increases the receptive fields of feature points without decreasing the resolution of the feature maps, as claimed in [5]. This is the reason why we choose D-LinkNet as the starting point in the project.

C. Modifications on D-LinkNet

We can notice that the whole encoder (i.e. ResNet34) downsizes the image by a factor of 32. For a test image of size 608×608 in our case, the last layer of the encoder can yield a feature map of size 19×19. The aggregation of dilated convolutions can have a receptive field of size 2^{n+1} [5], where n is the number of dilated convolutions being aggregated. For $n = 4$ in the original D-LinkNet, the receptive field exceeds the region of the feature map in our case. Therefore, in the center part of the network, we put only three dilated convolutions with dilation rate 1, 2, 4. This is sufficient to globally extract semantic information from a given image.

D-LinkNet has three shortcut layers between encoder and

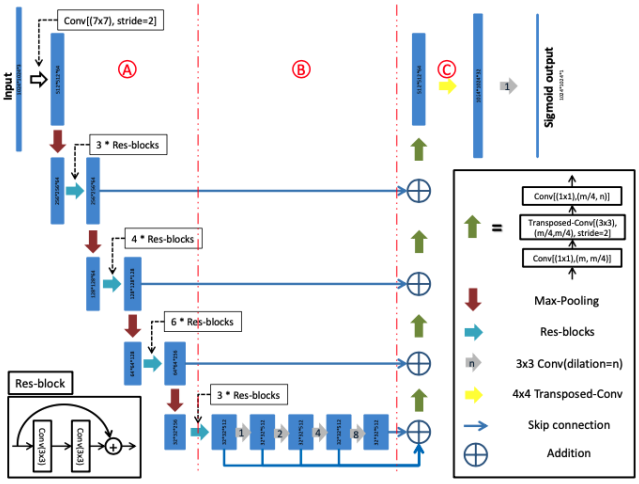


Figure 1: Architecture of D-LinkNet (figure from [1])

decoder. By taking computational efficiency into account, D-LinkNet follows the idea of Linknet that the shortcut layer is simply summing up the feature maps. Though this simplifies the network structure, it somehow imposes the challenge, that the output of the each block of encoder is shared by two distinct layers: one requires exploration of latent information, and another one at the decoder may need auxiliary spatial information. Concatenation of features like the shortcut in U-Net [2] is one way to merge the features, we can also learn the residual correction from the encoder to the decoder. Hence one convolution layer (with kernel size 3×3) is added at each shortcut of our new D-LinkNet (without changing the channel size). For convenience in the later discussion, we call this modified network architecture as D-LinkNet+.

We can also add additional shortcut layer at the shallow part of the encoder. This may not improve the performance since the shallow layer may only hold low-level information.

To have a good initial estimate of network parameters, we use the ResNet34 pretrained from ImageNet [6]. Just like ResNet [3], in the whole network, batch normalization [7] is applied after every linear operation. This helps to properly re-adjust the feature distribution before the nonlinear activation (ReLU [8] in our case). A sigmoid function is placed at the end of the network, which is to compress the output to be in range $[0, 1]$.

III. Loss Function

Given an image as the input, our network outputs a confidence map $\hat{\mathbf{M}}$ having the same height and width as the input. During the training, provided with a ground truth mask $\mathbf{M}$, binary cross entropy (BCE) is used to measure the deviation from the true segmentation:

$$L_{bce} = \frac{1}{m} \sum_{i,j} -M_{i,j} \log \hat{M}_{i,j} - (1 - M_{i,j}) \log \left(1 - \hat{M}_{i,j}\right)$$

where m is the total number of pixels in the image, $\hat{M}_{i,j} \in [0, 1]$ and $M_{i,j} \in \{0, 1\}$.

Minimizing BCE loss can improve the per-pixel classification, we can also maximize the F1 score $\frac{2 \sum_{i,j} M_{i,j} \hat{M}_{i,j}}{\sum_{i,j} M_{i,j} + \sum_{i,j} \hat{M}_{i,j}}$, which evaluates the segmentation from a global perspective. This is almost equivalent to minimizing the dice loss:

$$L_d = 1 - \frac{1 + 2 \sum_{i,j} M_{i,j} \hat{M}_{i,j}}{1 + \sum_{i,j} M_{i,j} + \sum_{i,j} \hat{M}_{i,j}}$$

We combine the loss in the formulation, just same as in [1]:

$$L = L_{bce} + L_d \tag{1}$$

During the training, we minimize the average of the combined loss L among all the images in the batch.

IV. Data Augmentation

Due to the limited training data, data augmentation is the key to prevent the overfitting. The simplest way to augment training images is to randomly do horizontal flip, vertical flip and rotation by 90 degrees. Each specific operation occurs by a 50% probability, hence the image can be augmented by 8 times. For convenience of discussion, we denote this augmentation method as DA8.

A. Random rotation

Since most roads in the training data are either horizontal or vertical, training with those images can incur potential overfitting, even if the image has been flipped or rotated by 90 degrees. To cope with this issue, we deploy random rotation such that the image coordinate frame is anti-clockwise rotated by some angle θ, uniformly sampled from the interval $[0, \frac{\pi}{2})$. The rotation of the image frame can be simply implemented by the 2D affine transformation:

$$\mathbf{R}(\mathbf{x}' - \mathbf{c}') = \mathbf{x} - \mathbf{c} \implies \mathbf{R}\mathbf{x}' + \mathbf{c} - \mathbf{R}\mathbf{c}' = \mathbf{x} \tag{2}$$

where $\mathbf{R} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}$. $\mathbf{x}$ and $\mathbf{x}'$ are the 2D image coordinates of the original image and rotated image. $\mathbf{c}$ and $\mathbf{c}'$ correspond to the image center of the original image and rotated image. The equation (2) is obviously an inverse mapping. So, given a coordinate $\mathbf{x}$, cubic interpolant is used to interpolate the intensities from the original image. The border reflection is also used to symmetrically fill the unknown region. As shown in figure 2, our random rotation can yield a realistic image. Especially when the roads are either horizontal or vertical, the border reflection can smoothly extrapolate (or extend) the roads and houses.

B. Color transformation

The deep neural network is very sensitive to the change of input, where an extreme example can be adversarial attacks [9]. To make our augmented data having more independence, we transform the images into HSV color space and add some

(a) Original image (b) Rotated image with $\theta = \frac{\pi}{6}$

Figure 2: Original image and rotated image

random noise on those three channels. In our project, we set the random noise drawn from uniform distribution within the range $[-10, 10]$.

V. Implementation

Training setup: The network is implemented in PyTorch framework [10]. All experiments are done in Colab, where the training is assisted by NVIDIA T4 GPU. The training images are resized to 384×384, in order to allow downsizing the image exactly by a factor of 32. For random rotation, resizing is not needed since we can directly set the size of the output image, i.e. adjust image center $\mathbf{c}'$. Images are also normalized so that the intensities stay in range $[0, 1]$.

Optimization: We use mini-batch stochastic gradient descent to minimize the loss, where AMSGrad optimizer [11] is chosen in this project. AMSGrad is a variant of Adam [12], and has been successfully proved for its convergence. To avoid overfitting, we put L2 regularization on the weights, i.e. weight decay is used.

Test: For the provided test data, the test image is of size 608×608, which is a factor of 32. So, the test images are not resized. Given an output confidence map $\hat{\mathbf{M}}$, we use the threshold 0.5 to generate a binary segmentation mask. Test time augmentation (TTA) is applied as an alternative choice, such that the test image is augmented by DA8. In this way, 8 predictions (the confidence map) can be made for each test image. The output masks are finally transformed back to the original coordinate frame, i.e. inversely flipped and rotated. Those predictions are finally merged into one prediction mask by extracting the medians.

Evaluation: In this segmentation contest, we are required to submit the predicted labels for every 16×16 image patch from the test data. After our model outputs a binary mask, a 16×16 patch is cropped from it, labelled as the road (1) if the occupation ratio of white pixels on the patch exceeds the threshold 0.25; otherwise, labelled as the background (0). This is equivalent to produce a downsized binary mask. As required in the competition, F1 score is used to evaluate the performance of segmentation.

VI. Experimental Results

In all experiments, the batch size is set as 20, and the learning rate is 10^{-4}. For L2 regularization, the weight decay is set as 10^{-5}. The number of epochs is set in the way that the final loss can just settle down to a small value, e.g. 0.05 During the training, we use all the training data, including 100 satellite images. During test, we apply our network to the provided 50 test images and obtain the F1 score from the AICrow platform.

A. Main experiments

In our main experiment, we do the ablation study on D-LinkNet+, and compare its performance with D-LinkNet.

	DA8	Rot	HSV	TTA	Epochs	F1
D-LinkNet+	-	-	-	-	300	0.874
D-LinkNet+	✓	-	-	-	500	0.904
D-LinkNet	✓	✓	-	-	1500	0.917
D-LinkNet	✓	✓	-	✓	1500	0.919
D-LinkNet+	✓	✓	-	-	1500	0.919
D-LinkNet+	✓	✓	-	✓	1500	0.921
D-LinkNet+	✓	✓	✓	-	1500	0.921
D-LinkNet+	✓	✓	✓	✓	1500	**0.922**

Table I: Evaluation of various models on the test data (Rot indicates random rotation and HSV indicates the random perturbation on HSV channels)

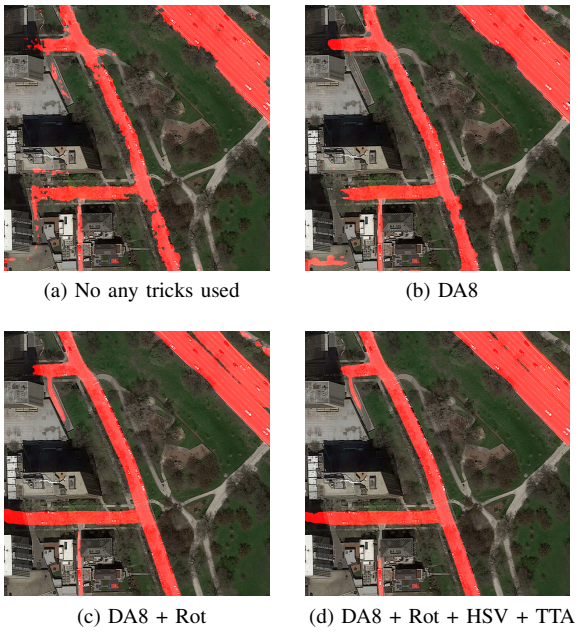
As shown in the table I, for both D-linkNet and D-LinkNet+, TTA can only make marginal improvement between 0.001 and 0.002. For the data augmentation, DA8 makes largest contribution to the performance, e.g. 0.03 for D-LinkNet+. Random rotation also plays an important role, which increments 0.015 to the F1 score of D-LinkNet+. And random perturbation on HSV color space further increases the performance by 0.002.

Another remark is that, when only comparing the best results, D-LinkNet+ outperforms D-LinkNet by 0.003, which empirically implies that our new network architecture may have greater expression power. Unsurprisingly, the highest F1 score 0.922 is finally achieved by D-LinkNet+. This model with highest score is chosen to be our final submission.

The ablation study of D-LinkNet+ can be visualized in the figure 3, with the road segmentation shown as a red mask. The selected test image is a typically hard example, where the walking paths have spanned almost half of the image. An incredible fact is that D-LinkNet+ can still realize that the walking paths are not roads, even no data augmentation is applied, as shown in figure 3(a). This may be explained by the use of aggregated dilated convolution. We can also see that the segmentation boundary becomes sharper once more data augmentation is applied.

B. Other experiments

Focal loss: The focal loss [13] penalizes relatively more on hard samples. This may benefit our network to



(a) No any tricks used (b) DA8

(c) DA8 + Rot (d) DA8 + Rot + HSV + TTA

Figure 3: Ablation study on test result of D-LinkNet+

overcome some hard cases like the detection of shadowed roads. In the additional experiment, we replace BCE loss with focal loss in equation (1). When only DA8 and random rotation are applied, the F1 score of the trained D-LinkNet+ is 0.917, which does not give extra improvement. A reason for this may be that some of the hard samples (i.e. 'unconfident' ones) with more penalization may be incorrectly labelled pixels.

ResNet18: Lighter network architecture, intuitively, should have more possibility to avoid overfitting. To test this hypothesis, we replace ResNet34 with ResNet18 in D-LinkNet+. Note that ResNet18 has almost half number of parameters as ResNet34. The result is not promising, that the new performance is degraded to 0.909 (only DA8 and random rotation are applied). As a result, road segmentation is thought as a complex task, and hence needs deeper inference on features.

Learning rate decay: Reducing the step size during optimization may help with convergence, especially for stochastic gradient descent. In another experiment, we decay the learning rate after every 500 epochs by multiplying a ratio of 0.4. However, there is no improvement on performance, which may be explained by the fact that AMSGrad can already adaptively adjust the step size.

Random seed and reproducibility: Another frustrating fact is that the segmentation performance can be easily influenced by the random seed. For instance, by using all data augmentation methods (except TTA), the F1 score of D-LinkNet+ can accidentally drop from 0.921 to 0.918. At last, we decide to fix the random seed as 2019 to both Numpy and Torch. But we still cannot guarantee the reproducibility for the models, unless we set *torch.backends.cudnn.deterministic = True* which may have negative impact on training.

Ensemble: To struggle with increasing the F1 score, we use the ensemble as a test strategy. The ensemble here means to simply compute the weighted average of the predictions from various models, e.g. 0.4 D-LinkNet+, 0.35 D-LinkNet and 0.25 LinkNet in our implementation. In the experiment, unfortunately, the use of ensemble does not show any improvement.

VII. Discussion

There are many unsolved issues in the road segmentation. For example, the detection of shadowed roads remains as a challenge. One possible improvement can be made by further modifying the network architecture. One way can be adding more dilated convolutions in the shortcut layer, which may nevertheless incur more computational cost. Concatenation of features in the shortcut is also worth trying out, like the shortcut in U-Net. Since most roads are straight, we may consider putting some constraints in the loss formulation, or trying some other kind of loss, like boundary loss [14]. Post-processing may further improve the performance, like conditional random field.

An obvious downside in our project is that we do no have a validation dataset. Since the given training dataset is very small, we did not separate it into validation and training data. We may have to find some external satellite images, like the Chicago dataset from [15]. A good validation dataset can help evaluate the model performance, which can be even used for determining an early stop during training. Note that the use of early stop may be possible to avoid performance degrading caused by overfitting.

Though our D-LinkNet+ can reach a good score of 0.921 in the road segmentation, the model is still used as a black box. Playing with a black box hence becomes an empirical task. In the future, we may want to use some traditional works to replace the unpredictability of using neural network for image segmentation.

VIII. Acknowledgement

Our work is strongly based on the previous work of D-LinkNet [1], including the network architecture and loss design. To Colab, really thanks for its free online GPU that reduces uncountable effort in training.

References

[1] L. Zhou, C. Zhang, and M. Wu, "D-linknet: Linknet with pretrained encoder and dilated convolution for high resolution satellite imagery road extraction." in *CVPR Workshops*, 2018, pp. 182–186.

[2] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[4] A. Chaurasia and E. Culurciello, "Linknet: Exploiting encoder representations for efficient semantic segmentation," in *2017 IEEE Visual Communications and Image Processing (VCIP)*. IEEE, 2017, pp. 1–4.

[5] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[7] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[8] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011, pp. 315–323.

[9] S. Huang, N. Papernot, I. Goodfellow, Y. Duan, and P. Abbeel, "Adversarial attacks on neural network policies," *arXiv preprint arXiv:1702.02284*, 2017.

[10] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[11] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of adam and beyond," *arXiv preprint arXiv:1904.09237*, 2019.

[12] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[13] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[14] H. Kervadec, J. Bouchtiba, C. Desrosiers, É. Granger, J. Dolz, and I. B. Ayed, "Boundary loss for highly unbalanced segmentation," *arXiv preprint arXiv:1812.07032*, 2018.

[15] P. Kaiser, J. D. Wegner, A. Lucchi, M. Jaggi, T. Hofmann, and K. Schindler, "Learning aerial image segmentation from online maps," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 11, pp. 6054–6068, 2017.