CS230

Final Report: Deep-learning for Road Segmentation

Reese Guo
Department of Civil and Environmental Engineering
Stanford University
shijieg@stanford.edu

Tianheng Shi
Department of Mechanical Engineering
Stanford University
ts5@stanford.edu

Yiting Zhao
Department of Management Science and Engineering
Stanford University
yitzhao@stanford.edu

Abstract

This project aims to tackle the problem of road segmentation from satellite images. Four encoder decoder deep learning architectures, including UNet, ResUNet, ResUNet++, and UNet-GAN, are implemented for this task. These architectures' performances, measured by IoU score, are compared. Among the four models, ResUNet yields the best performance for having the smoothest and least-noisy predictions, followed by UNet-GAN for its ability to accurately predict road width. UNet and ResUNet++ also yield satisfactory predictions.

1 Introduction

Road segmentation plays an important role for urban planning, which requires frequent and accurate update of map. It can also contribute to emergency evacuation, traffic planning, and many other purposes. Extensive researches have been conducted on efficient and accurate road segmentation deep learning algorithms (Abdollahi et al., 2020). Although deep learning techniques have achieved impressive accuracy in some applications, the current methods still suffer from problems including, 1) lack of segmentation consistency with various road types, and 2) lower segmentation accuracy when occlusions are present, especially for curved roads (Gao et al., 2019; Abdollahi et al., 2020). This study aims to tackle the challenge of segmenting road features from satellite images and improve existing models to solve the aforementioned challenges.

2 Related Work

UNet model by Ronneberger et al. (2015) provided great performances for image segmentation tasks. It can be seen as an Encoder-Decoder structure with shortcuts of copy and crop. Such shortcuts can combine the high resolution information with lower resolution features for the upsampling decoder to use. Zhang et al. (2018) proposed ResUNet utilizing residual connections in U-Net. On the base of UNet, ResUNet provides residual units for both encoder and decoder parts, which can both facilitate the information propagation and ease the model training. In addition, Jha et al. (2019) provided a variation of ResUNet called ResUnet++. It inserted squeeze layers and excitation layers into the encoder structure and attention layers into the decoder structure. Such additions can improve the representative power of the ResUNet model. Since UNet model failed to segment small roads in parking lots, Filin et al. (2018) further refined the model by processing the road pixels to fill the gaps. Sun et al. (2018) further generated road maps by stacking two U-Nets. Haeyun et al. (2021) placed SPP at the end of the encoder of U-Net to aggregate multi-scale contextual information. Inspired by the effectiveness of dense convolutions for feature learning and residuals, Eerapu et al.

CS230: Deep Learning, Spring 2022, Stanford University, CA.

(2019) proposed a dense refinement residual network (DRR Net) for semantic segmentation, which is composed of multiple DRR modules for the extraction of diversified roads alleviating the class imbalance problem.

3 Models

We have implemented our baseline model UNet with pytorch framework, which is proposed by Ronneberger et al. (2015). Based on the implementation of pytorch UNet, the team explored existing variations of UNet model for comparison. ResUNet++ and ResUNet were implemented also with pytorch framework. To improve these existing models, the team made use of the technology of Generative Adversarial Network model for better segmentation image generations. Models details are discussed in the following subsections.

3.1 UNet

In order to use the available annotated samples more efficiently, Ronneberger et al. (2015) proposed UNet, which consists of a contracting path to capture context and a symmetric expanding path for precise localization. The architecture is illustrated in Fig.5 in Appendix.B.

The left side is the contracting path which has repeated implementation of two 3×3 unpadded convolutions, followed by a rectified linear unit and a down-sampling layer of 2×2 max pooling operation with stride of 2. At each step, the number of feature channels is doubled. The right side is the expanding path, which consists of an upsampling of the feature map, a 2×2 up-convolution, a concatenation layer which cropped feature map from the left side correspondingly, and two 3×3 unpadded convolutions, finally followed by a ReLU layer. The final layer is a 1×1 convolution that maps the feature vector to the desired number of classes.

3.2 ResUNet

Adding more hidden layers to the neural networks may improve the performance while hamper the training. Therefore, He et al. (2016) proposed the residual neural network to address this problem. The proposed architecture consists of a series of stacked residual units, which can be illustrated as

$$\begin{aligned} y_l &= h(x_l) + \mathscr{F}(x_l, W_l) \\ x_{l+1} &= f(y_l) \end{aligned}$$

where x_l and x_{l+1} are the input and the output of the l-th residual unit, $h(\cdot)$ is a identity mapping function, $\mathscr{F}(\cdot)$ is the residual function, and $f(\cdot)$ is the activation function.

To combine the strengths of both UNet and residual neural networks, Zhang et al. (2018) further proposed ResUNet. The architecture is illustrated in Fig.6 in Appendix.B. The network comprises of three parts: encoding, bridge and decoding. It first encodes the input image into compact representation, then recovers the representations to a pixel-wise categorization, with a middle bridge connecting the two parts. Unlike the UNet, the ResUNet are built with residual units which consists of two 3×3 convolution blocks and an identity mapping, where each convolution block includes a Batch Normalization layer, a ReLU activation layer and a convolutional layer.

3.3 ResUNet++

By further taking advantage of the squeeze-and-excitation block and the attention block, Jha et al. (2019) proposed the ResUNet++ architecture. In the encoding part, the output is passed through the squeeze-and-excitation block to boost the representative power of the network, where each channel is first squeezed by using global average pooling for generating channel-wise statistics and then channel-wise dependencies are captured fully by the excitation part. In the decoding part, the attention block is applied before each unit to increase the effectiveness of feature maps, since attention mechanism determines which parts of the network require more attention in the neural network.

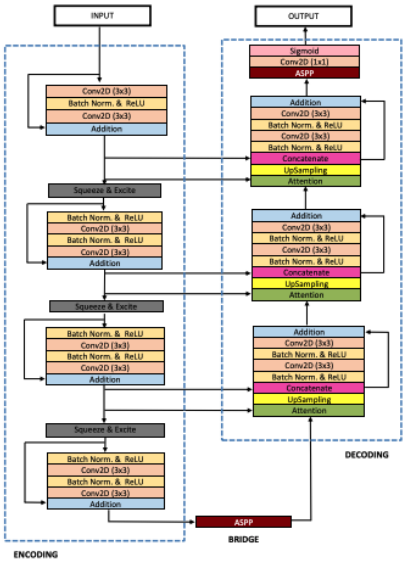

Figure 1: ResUNet++ Architecture

3.4 GAN

The generative adversarial networks (GAN) architecture consists of a generator and a discriminator. The generator model uses the same structure as the baseline UNet Model. The discriminator model is based on the discriminator structure in Deep Convolutional Generative Adversarial Networks (DCGAN) architecture as proposed by Radford et al. (2015). The generator aims to extract road features from satellite images and the discriminator tries to distinguish the generator prediction from ground-truth road features. The discriminator consists of three sets of strided two-dimensional convolution layers, batch norm layers, and LeakyReLU activations. The discriminator convolution layers are followed by a fully-connected layer and sigmoid activation function for output. The input to the discriminator is a $256 \times 256 \times 1$ image and the output is a real-value probability indicating the probability that the input image is real image.

4 Experiments

Github Link: `https://github.com/TianhengShi/cs230_project`

4.1 Dataset and Features

The dataset this project uses is the Massachusetts Roads Dataset, which is accessible for public, made by Mnih (2013). The datasets consist of 1108 training images, 14 validation images, and 49 testing images. Each image has spatial resolution of 1m and shape of $1500 \times 1500 \times 3$ pixels. The primary features used are the RGB channels of each image. Labels can treated as binary classification labels, which represent either backgrounds or roads.

To speed up learning and to create more training examples, we subsample 256×256 pixel images from each original training image. We also randomly apply horizontal flips, vertical flips, and 90-degree rotations to the subsampled images to generate augmented data for the training set. Even though the team tried to use to original size for validation, the limited RAM space made it impossible. Same subsample method was done to the validation dataset for consistence.

4.2 Evaluation Metrics and Loss Function

Since the majority of pixels in a given satellite image is not road features, our dataset suffers from data imbalance problem. To offset the influence of data imbalance on model prediction ability, we chose to use the dice loss function as our loss function of our deep learning architecture. The dice

loss function can be expressed as,

$$L_{\text{dice}} = 1 - \frac{2\sum_{i=1}^{N} p_i g_i}{\sum_{i=1}^{N} p_i^2 + \sum_{i=1}^{N} g_i^2},$$

where N is the total number of pixels in an image, and p_i and g_i are the pixel values for the i^{th} pixel in the prediction image and ground-truth image respectively.

The loss functions used for GAN is changed into the following equations:

$$L_D = -\left(\log(D(x)) + \log(1 - D(G(z)))\right)$$
$$L_G = L_{\text{dice}} - \log(D(G(z)))$$

where L_D and L_G are the loss functions for the discriminator and generator respectively, $D(x)$ is the discriminator output for a real instance, $D(G(z))$ is the discriminator output for a fake instance generated by the generator.

To quantitatively evaluate the performance of the model, we use Intersection over Union (IoU) metric to measure the model's prediction accuracy. The IoU measure is defined as,

$$\textit{Intersection over Union (IoU)} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}},$$

The IoU measure ranges from 0 to 1. The closer the measure is to 1, the better the prediction accuracy.

4.3 Experimental Details

With small batch size of 4, Stochastic Gradient Descent optimizers are used for U-net, ResUNet, and ResUNet ++ architectures. For the UNet architecture, a starting learning rate of 0.001 is used, with learning rate decays to 0.0005 after 10 epochs and to 0.0001 after 20 epochs. The UNet architecture is trained for 50 epochs. For the ResUNet and ResUNet++ architectures, a constant learning rate of 0.0005 is used. ResUNet is trained for 40 epochs and ResUNet++ is trained for 60 epochs.

For the generative adversarial model, Stochastic Gradient Descent optimizers are used with a starting learning rate of 0.001. Learning rate decay with a factor of 0.5 for every 15 epochs is implemented with the GAN optimizers. The number of training epochs for the GAN model is 45 epochs. For UNet, ResUNet, ResUNet++, and the generator in GAN, Kaiming Initializations are being applied.

5 Results and Discussions

We trained our U-net architecture as mentioned in Section 3.1 and obtain our baseline model. Figure 2 below shows the training loss for the baseline UNet after 40 epochs of training.

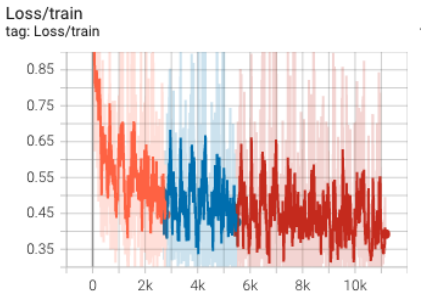

Figure 2: Training Loss vs. Iterations for UNet

Figure 3 and 4 shows a visualization of a target and prediction pair. Based on the results metric and loss, as shown in Table 1, it can be observed that UNet and ResUNet++ all generated similar results both qualitatively and quantitatively. ResUNet performs better than UNet and ResUNet++, with smoother segments that can be visually observed as in Fig 3. Possible reason for this situation could be that the residual block can guarantee that ResUNet performs as good as UNet. The similar results from UNet and ResUNet++ could be explained by several reasons. First, due to hardware

limitations, all data images were downsized into smaller dimensions. With downsized images, further addition into the UNet model could lead to little improvement effect. Similarly, the batch size used for all training process was limited to 4 due to VM limitations. Such a small batch size enforced the optimizer to be SGD. Both the small batch size and SGD optimizer generated stochasticity into the training.

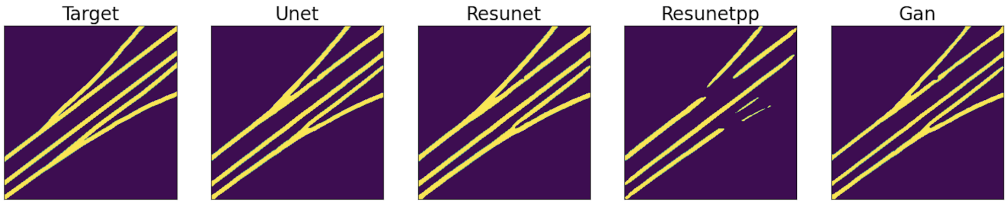

Figure 3: Comparison of models predictions to target image.

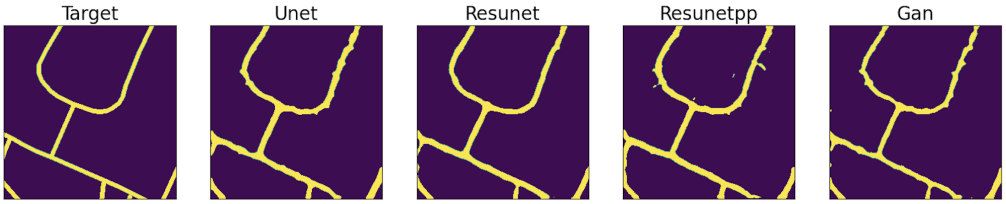

Figure 4: Comparison of models predictions to target image.

Architectures	Training Loss	Validation Loss	Training IoU	Validation IoU
UNet	0.4248	0.4355	0.6822	0.6794
ResUNet	0.4107	0.4215	0.6990	0.6918
ResUNet++	0.4145	0.4437	0.6954	0.6744
UNet-GAN	1.2730	1.3150	0.6956	0.6824

Table 1: Training and validation results for UNet, ResUNet, ResUNet++, and UNet-GAN architectures.

For GAN, performance improvement is observed. During the training process, resultant model weights from UNet was loaded as pretrained weight for the generator. This trick lifted the performance of generator at the beginning of the training process and facilitated the training. The existence of the discriminator changed the loss function for the generator compared with UNet. Such an existence would force the generator to generate result images that are similar to target pixel-wise, with less noisy segmentations. From Fig 4, it can be observed that the width of the road segments is closer to the ground truth. All other models' results have apparently wider road segments. Such an improvement also shows how the discriminator could help in this case.

6 Future Work

From section 5, it is concluded that ResUNet and GAN out-perform UNet and ResUNet++. Since ResUNet generates smoother and less-noisy outputs and GAN can better predict road width, one possible improvement is to use ResUNet as the generator structure for a new GAN architecture. Since our current results suffers from data imbalance problem, future work can including implementing multiple DRR modules as suggested by Eerapu et al. (2019) to extract diversified roads. Additionally, as our goal is to optimize IoU, we may also try to revise our current loss function. Berman and Blaschko (2017) proposed the Lovász-Softmax loss, which is a method for direct optimization of the mean intersection-over-union loss in neural networks. In our future implementation, we may use a composite loss which is composed of both the Lovász-Softmax loss and our current loss.

References

Abdollahi, A., B. Pradhan, N. Shukla, S. Chakraborty, and A. Alamri (2020). Deep learning approaches applied to remote sensing datasets for road extraction: A state-of-the-art review. *Remote Sensing 12*(9), 1444.

Berman, M. and M. B. Blaschko (2017). Optimization of the jaccard index for image segmentation with the lovász hinge. *CoRR abs/1705.08790*.

Eerapu, K. K., B. Ashwath, S. Lal, F. Dell'Acqua, and A. V. Narasimha Dhan (2019). Dense refinement residual network for road extraction from aerial imagery data. *IEEE Access 7*, 151764–151782.

Filin, O., A. Zapara, and S. Panchenko (2018, June). Road detection with eosresunet and post vectorizing algorithm. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Gao, L., W. Song, J. Dai, and Y. Chen (2019). Road extraction from high-resolution remote sensing imagery using refined deep residual convolutional neural network. *Remote sensing 11*(5), 552.

Haeyun, L., K. Lee, J. Kim, Y. Na, J. Park, J. Choi, and J. Hwang (2021, 03). Local similarity siamese network for urban land change detection on remote sensing images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing PP*, 1–1.

He, K., X. Zhang, S. Ren, and J. Sun (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778.

Jha, D., P. H. Smedsrud, M. A. Riegler, D. Johansen, T. de Lange, P. Halvorsen, and H. D. Johansen (2019). Resunet++: An advanced architecture for medical image segmentation.

Mnih, V. (2013). *Machine learning for aerial image labeling*. University of Toronto (Canada).

Radford, A., L. Metz, and S. Chintala (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*.

Ronneberger, O., P. Fischer, and T. Brox (2015). U-net: Convolutional networks for biomedical image segmentation. *CoRR abs/1505.04597*.

Sun, T., Z. Chen, W. Yang, and Y. Wang (2018, June). Stacked u-nets with multi-output for road extraction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Zhang, Z., Q. Liu, and Y. Wang (2018). Road extraction by deep residual u-net. *IEEE Geoscience and Remote Sensing Letters 15*(5), 749–753.

Appendices

A Contributions

Tianheng contributed to general framework setup, data preprocessing UNet and GAN models implementation and tuning, report, powerpoint presentation.

Yiting contributed to the general framework setup, ResUNet and ResUNet++ models implementation and tuning, report and powerpoint presentation.

Reese contributed to the general framework setup, data preprocessing, implementation and tuning of the UNet and GAN architectures, report, and powerpoint presentation.

B Models

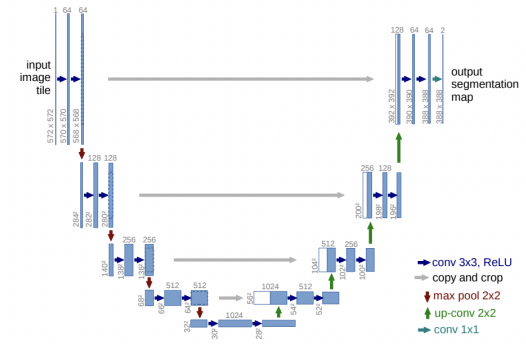

Figure 5: UNet Architecture

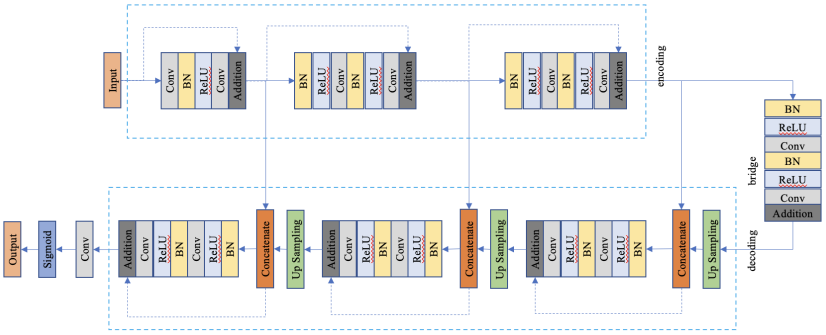

Figure 6: ResUNet Architecture