Semantic segmentation optimization analysis of remote sensing image

Jinwu Wang
Australian National University
Canberra
u7354172

Abstract

Semantic segmentation of remote sensing images is to classify the types of surface features at the pixel point level. In order to achieve accurate semantic segmentation of surface features in high-resolution remote sensing images, a model optimization method based on FCN network for semantic segmentation of remote sensing images is proposed. The model is optimized by adding Batch Normalization Layer to prevent overfitting, using Focal loss loss function to solve the sample imbalance problem, and using Sobel edge extraction operator to solve the edge detection problem. Finally, the optimized semantic segmentation results are obtained, and the accuracy is improved to 87.04%, and the average cross-merge ratio is improved to 47.44%, comparing with SVM, FCN32s, FCN16s and U-net network to show the advantages of the model. Code is at: https://github.com/WJ-Fifth/Semantic-Segmentation-of-Remote-Sensing-Images-Based-on-FCN

1 INTRODUCTION

With the continuous innovation and development of various high-resolution remote sensing satellites, it has become easy to acquire remote sensing images, leading to the generation and application of a large number of high-resolution remote sensing images. Among them, automatic segmentation and extraction of surface information from high-resolution remote sensing images, as an important technique of ground exploration, has long been widely used in various fields such as surface mapping, mineral exploration, urban planning, forestry measurement and disaster assessment.

The introduction of deep learning has made a breakthrough in the classification segmentation of large-scale massive data, and the automated remote sensing interpretation algorithm based on deep learning has greatly accelerated the speed of earth observation and reduced the economic cost of field survey.

In 2014, Jonathan Long, Evan Shelhamer, and Trevor Darrell at UC Berkeley proposed a fully convolutional neural network (FCN) [1] based on CNN. The model removes the fully connected layer at the end and uses a deconvolutional layer for upsampling operations to implement pixel point segmentation as well as classification methods. The FCN semantic segmentation method is used to build classifiers for better quality semantic segmentation.

The main analysis of this project is based on the optimization of the algorithm to improve the performance of the FCN network model, and then the final model can be more adapted to the semantic segmentation experiments of high-resolution remote sensing images. From optimizing the model by adding Batch Normalization Layer; optimizing the loss function by using Focal loss to solve the class imblance problem; and using the edge detection algorithm Canny to improve the edge detection capability of the model.

This project uses the open source Gaofen Image Dataset (GID) from Wuhan University, which has a large coverage, wide distribution, and high spatial resolution, and outperforms existing land cover datasets. The GID consists of two parts: a large-scale classification set and a fine-grained land cover classification set. The large-scale classification set contains 150 pixel-level annotated GF-2 images, based on a collection of training and validation images from 5 classes [2], and the format of the dataset in this study is modeled after the Segmentation Image Sets dataset format in the PASCAL Visual Object Class Challenge 2012 (VOC2012) to set up the image segmentation dataset [3].

2 Related work

2.1 Backbone-VGGNet

This project used VGG16 Net as the backbone of FCN network, because VGG16 puts forward the concept of feeling wild for the first time, using smaller convolutional kernels instead of large convolutional kernels, which can effectively avoid the problem of too many parameters while increasing the depth of the network [4] [5], and improve the model accuracy and model running speed.

receptive field, RF

In convolutional neural networks, determining the size of the region of the input layer corresponding to an element in the output of a given layer is called the perceptual field [4]. That is, a pixel on the output feature map (Feature map) of each layer of the convolutional neural network corresponds to the size of the region mapped to the input layer. By stacking several small convolutional kernels to accomplish the function of large convolutional kernels, we can save the parameters needed for model training while having the same perceptual field as large convolutional kernels.

Backbone structure

The up-sampling layer of this FCN model is implemented using a deconvolution layer. The deconvolution layer is the opposite of the convolution layer, where a pixel is transformed into a feature matrix by the deconvolution operation.

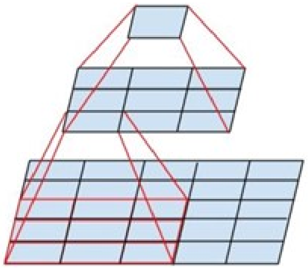


Figure 1: **receptive Field diagram**,

2.2 Up-sampling

Semantic segmentation requires pixel-level classification operations on remote sensing images, so the fully-connected layer cannot be used for overall image classification; instead, the nput Feature Map is reduced to an output feature map of the same size as the original feature map by up-sampling after the feature extraction is completed by the convolution layer.

The up-sampling layer of this FCN model is implemented using a deconvolution layer. The deconvolution layer is the opposite of the convolution layer, where a pixel is transformed into a feature matrix by the deconvolution operation.

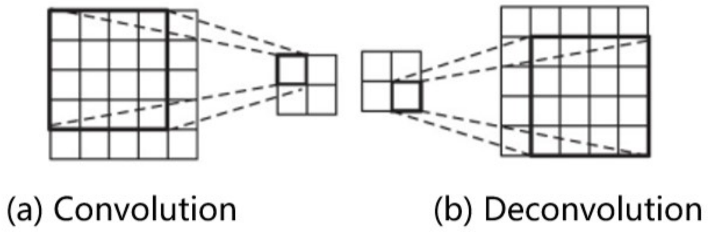


Figure 2: **Convolution & Deconvolution**,

2.3 Multi-scale feature fusion

In semantic segmentation, feature maps of different scales have different information. Low-level features contain more location and detail information and are suitable for learning local features. However, they are less semantic and noisier due to the smaller perceptual domain. High-level features have more abstract semantic information, but have very low resolution and poor perception of details. Fusing multiple features of different scale sizes is an important means to improve segmentation performance and model accuracy.

In FCN model that deconvolution results are combined with max-pooling layer corresponding to backbone to realize the fusion of feature maps of different scales. Combine the output feature maps of different deconvolution layers to get the final output label image and improve the local feature extraction capability of the model. FCN-8s multi-scale feature fusion is used in this paper, and the algorithm flow is as follows1:

```
Algorithm 1: FCN-8s
Input: The Max-pooling layers' outputs in backbone: $x_i$
Output: $8\times$ upsampled prediction
$score = DeConv(x_5)$;
$score = DeConv(x_4 + score)$;
$score = DeConv(x_3 + score)$;
return score
```

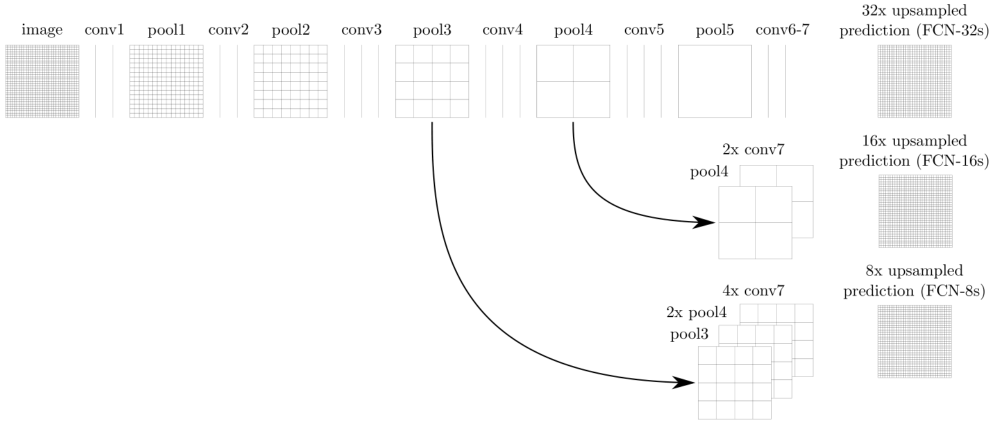


image
conv1
pool1
conv2
pool2
conv3
pool3
conv4
pool4
conv5
pool5
conv6-7
32x upsampled prediction (FCN-32s)
2x conv7
pool4
16x upsampled prediction (FCN-16s)
4x conv7
2x pool4
pool3
8x upsampled prediction (FCN-8s)

Figure 3: **Multi-scale feature fusion**,

3 Methods

3.1 Batch Normalization

FCN model in order to be able to obtain semantic level segmentation results. After using the deconvolution layer instead of the fully connected layer, it is not possible to prevent the model overfitting and non-convergence problems using the method of random deactivation of neurons (dropout), and this project used the BN (Batch Normalization) layer instead of the Dropout layer. To speed up the training speed; to improve the model generalization and convergence ability as well as to improve the model accuracy [6].

According the the Algorithm 2, the caculation of the Batch Normalization layer as follows:

(1) Calculate the mean of batch data: μ;

(2) Calculate the variance of batch data: σ^2;

(3) Standardized processing of sample data;

(4) Translate and scale the feature matrix, train the two parameters γ and β, and realize the feature extraction of the model.

Algorithm 2: Batch Normalization

```
Input: Values of $x$ over a mini-batch: $\mathcal{B} = \{x_1, x_2, \ldots, x_m\}$
       Patameters to be learned: $\gamma, \beta$
Output: $y_i = BN_{\gamma,\beta}(x_i)$
$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i$ //mini-batch mean;
$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2$ //mini-batch variance;
$\widehat{x_i} \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \varepsilon}}$ //normalize;
$y_i \leftarrow \gamma\widehat{x_i} + \beta \equiv BN_{\gamma,\beta}(x_i)$ //scale and shift;
```

3.2 Class imbalance

The presence of a large difference in the number of training samples from different classes in a classification task is called "class imbalance". When the amount of one sample is much higher than the others, the prediction result of the learned model will be greatly biased towards that sample; or if the amount of one sample is much lower than the others, the model will not be able to make predictions and the network learner will be useless.

The category imbalance problem is common in semantic segmentation, and the problem is more serious in remote sensing images, where the number of all categories in remote sensing images is uncertain, and there is a high possibility of having too much data in a single sample as well as the problem of missing single samples.

In this project, the model is trained by constructing a new loss function instead of the cross-entropy loss function in the original model. Cross-entropy, as the most commonly used pixel-level loss function in image semantic segmentation tasks, is generally used instead of the combination of the mean squared difference loss function and the sigmoid activation function.

$$CE(p_t, y) = \begin{cases} -\log(p_t) & \text{if } y = 1 \\ -\log(1 - p_t) & \text{otherwise} \end{cases} \tag{1}$$

However, the cross-entropy loss function can only classify samples according to their probabilities, and the weights of each training sample are the same. In the process of model training, model training usually favours a high percentage of samples. By improving the cross-entropy, the weights of easily classified samples are reduced, thus making the model more inclined to the hard-to-classify samples in the training process.

Focal loss [7] improves on the standard cross-entropy loss function. By reducing the weight of the easily classified samples, Focal loss pays more attention to the hard to classify samples during training Therefore, it solves the problem of severe imbalance of sample proportion in semantic segmentation.

A factor γ is added to the original one, where $\gamma > 0$ makes the loss of easy-to-classify samples reduced. Thus, the model focuses more on the difficult and misclassified samples. And a balancing factor α is added to balance the uneven proportions of various samples themselves [7].

$$FL(p_t) = -\alpha(1 - p_t)^{\gamma} \log(p_t) \tag{2}$$

3.3 Egde detection

Edge detection is the calculation, metric and localization of grayscale changes of edge pixels, which can extract image edge features.

Remote sensing images have a huge amount of pixels, the sample categories are close to each other, and the sample boundaries are not obvious, which causes the uncertainty of remote sensing image edges to be more serious compared with ordinary optical images. The main uncertainties are noise; inherent uncertainty of edges; scale uncertainty, etc.

The **Canny detection algorithm** [8] is used to calculate the image luminance function as a grayscale approximation. Using the Sobel filter at any point of the image will produce the corresponding grayscale vector or its normal vector. The pixel that reaches the extreme value is detected as an edge based on the grayscale weighted difference between adjacent pixel points. It can effectively smooth out noise and detect more accurate edge orientation.

(1) Image gradient values G are calculated by edge detection with horizontal and vertical Sobel convolution kernels, where x denotes each pixel in the 3×3-pixel matrix:

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * I = (I_3 + 2I_6 + I_9) - (I_1 + 2I_3 + I_5) \tag{3}$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} * I = (I_7 + 2I_8 + I_9) - (I_1 + 2I_2 + I_3) \tag{4}$$

$$G = \sqrt{G_x^2 + G_y^2} \tag{5}$$

(2) Then, the gradient direction can be calculated by G_x, G_y:

$$\theta = \arctan(\frac{G_y}{G_x}) \tag{6}$$

(3) Non-maximum edge contributing pixel points are suppressed by **Non-Maximum Suppression (NMS)** by ignoring edge points that do not contribute to feature visibility, labelling only the pixel points with maximum amplitude on the edge curve. Use the **Hysteresis Threshold** to retain pixels above the gradient amplitude and ignore pixels below the low threshold.

4 Experiments

4.1 Evaluation Criteria

In this study, the performance of the model on the GID-5 dataset is compared by **Accuracy** and **Mean Intersection over Union** to analyze the effect of the optimized model.

(1) Accuracy (Acc):

The percentage of pixels correctly predicted to be classified in the test set versus all pixels in the test set. where p_{ij} denotes the number of true values of i being predicted as j, and,

p_{ii} denotes the number of correctly predicted classified pixels.

$$Acc = \frac{\sum_{i=0}^{k} p_{ii}}{\sum_{i=0}^{k} \sum_{j=0}^{k} p_{ij}} \tag{7}$$

(3) Mean Intersection over Union (MIoU)

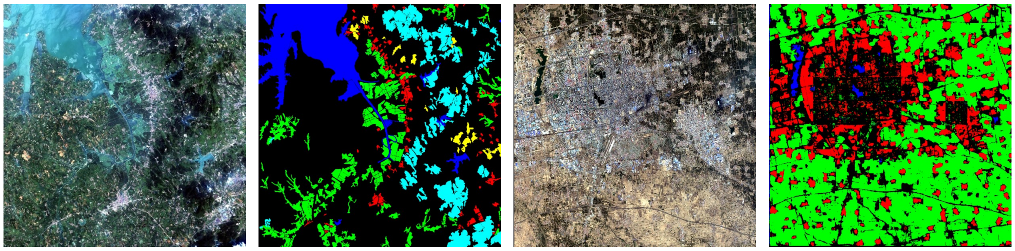
Calculate the ratio of the intersection and the union(IoU) of the sets of true and predicted values. p_{ij}, p_{ji} denotes false positive and false negative respectively.

Find the mean value of **IoU** for all classes. k denotes the total number of classes, and additionally counting the background as a class gives $(k + 1)$:

$$MIoU = \frac{1}{k+1} \left(\sum_{i=0}^{k} \left(\frac{p_{ii}}{\sum_{j=0}^{k} p_{ij} + \sum_{j=0}^{k} (p_{ji} - p_{ii})} \right) \right) \tag{8}$$

4.2 Comparison of experimental results

The merged images after edge extraction of the GID dataset using the Canny edge detection algorithm are used as input and fed into a new model optimized by Batch normalization and Focal loss to finally obtain semantic segmented images of the same size as the original images4.2, where red is a building, green is a farm, sky blue is a forest, yellow is a green field, and blue is a water area.



(a) real image 1 (b) label image 1 (c) real image 2 (d) label image 2

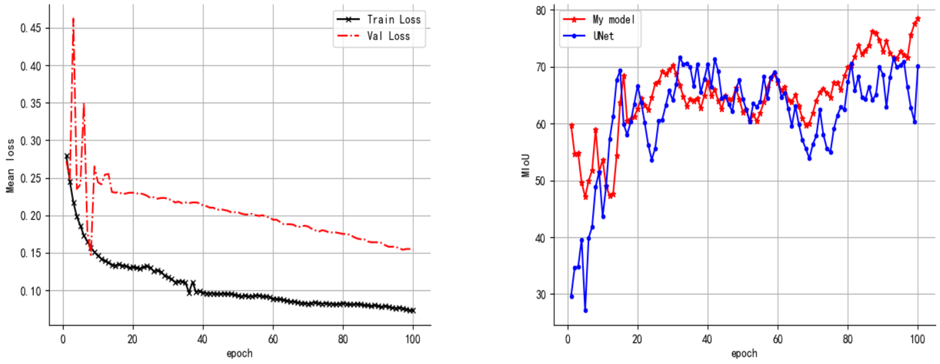
Figure 4: Predicted output label image

To verify the feasibility of this research method, the GID class-5 semantic segmentation dataset is trained with multiple semantic segmentation network models separately to derive the semantic segmentation results on the test set in different models. The experimental models are compared with other semantic segmentation models, such as SVM model, U-Net model, etc, FCN32s model and FCN16s model on the test dataset of GID for Acc and MIoU results4.2, respectively.

Model	**Acc**	**MIoU**
SVM	44.31%	–
FCN32s-CE	61.63%	34.37%
FCN16s-CE	68.41%	38.32%
FCN8s-CE	73.95%	41.31%
U-Net [9]	76.30%	42.87%
FCN8s-BN-FL	87.04%	47.44%

Table 1: Comparison of the results of multiple network models with the method of this paper

The training results of the dataset show that the performance of the FCN8s network optimized using Batch Normalization and Focal Loss is significantly improved with MIoU increase of 6.88%, and also has an advantage over the U-Net network practical for finely labeled images such as medical semantic segmentation. These improvements are mainly due to the optimization results of the Focal loss loss function for the class imbalance problem and the effective optimization operation of the dataset using the Canny edge detection operator, which effectively reduces the variability between different label classes and improves the accuracy as well as the performance of the model.



(a) Training and Validation loss vs. epochs

(b) My model and U-Net MIoU vs. epochs

Figure 5: Model training process comparison diagram

5 Conclusion

In this study, to achieve the semantic segmentation task of surface buildings in high-precision remote sensing images, this paper constructs an FCN network model, from using Batch Normalization to optimize the model to prevent overfitting; using Focal loss loss function instead of standard cross-entropy loss function to solve the category imbalance problem and using Canny edge extraction algorithm to remote sensing images The edge detection of remote sensing image dataset is carried out by using Canny edge extraction algorithm to realize data enhancement. Finally, the semantic segmentation performance of FCN model is improved.

References

[1] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014.

[2] Q. L. H. S. S. L. S. Y. L. Z. Xin-Yi Tong, Gui-Song Xia, "Land-cover classification with high-resolution remote sensing images using transferable deep models," *Remote Sensing of Environment, doi: 10.1016/j.rse.2019.111322*, 2020.

[3] M. Everingham, L. V. Gool, C. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, 2010.

[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[5] F. Haque, H.-Y. Lim, and D.-S. Kang, "Object detection based on vgg with resnet network," *2019 International Conference on Electronics, Information, and Communication (ICEIC)*, 2019.

[6] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *international conference on machine learning*, 2015.

[7] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[8] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1986.

[9] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *medical image computing and computer assisted intervention*, 2015.