U-net Network for Building Information Extraction of Remote-Sensing Imagery

https://doi.org/10.3991/ijoe.v14i12.9335

Jingtan Li, Maolin Xu(✉), Hongling Xiu
University of Science and Technology Liaoning, Anshan, China
ijt19950817110@163.com

Abstract—With the resolution of remote sensing images is getting higher and higher, high-resolution remote sensing images are widely used in many areas. Among them, image information extraction is one of the basic applications of remote sensing images. In the face of massive high-resolution remote sensing image data, the traditional method of target recognition is difficult to cope with. Therefore, this paper proposes a remote sensing image extraction based on U-net network. Firstly, the U-net semantic segmentation network is used to train the training set, and the validation set is used to verify the training set at the same time, and finally the test set is used for testing. The experimental results show that U-net can be applied to the extraction of buildings.

Keywords—High resolution remote sensing image, FCN; U-net, Building extraction

1 Introduction

In recent years, the word "artificial intelligence" has become more and more hot. From AlphaGo to machine translation, the application of deep learning methods has become more and more widespread. In 2006, Professor Geoffrey Hinton and others at the University of Toronto in Canada proposed the concept of deep learning. Up to now, the theory and methods of deep learning have developed rapidly. At present, in the field of close-range images, the method of deep learning can achieve better detection, segmentation and extraction effects on close-range images. With the continuous advancement of remote sensing image acquisition technology, how to effectively use the information in massive remote sensing data has become an urgent issue. Most of the traditional remote sensing information extraction methods rely on the combination of manual interpretation and computer processing. This requires not only the interpreter has rich geoscience knowledge, but also requires a lot of repetitive labor, and the method has low mobility [1-2]. The deep learning method allows the computer to automatically extract features without the need to manually design features, with strong generalization capabilities and good application prospects. Currently used deep learning architectures include convolutional neural networks (CNN), full convolutional networks (FCN), etc. This paper chooses U-net network architecture based on full convolutional network, and conducts feasibility study on whether it can be applied to large-scale and high spatial resolution remote sensing image information extraction. It uses Python language to implement programming on Pycharm. The extraction method is compared with the artificial intelligence end-to-end training method extraction efficiency, makeing the computer to automatically extract image features.

2 Extraction Algorithm

2.1 Convolutional Neural Network

As a classic deep learning model, convolutional neural network has been successfully applied in many fields such as image classification, speech recognition and machine translation[3-4]. In terms of image processing, compared with general neural networks, convolutional neural networks introduce local connections, weight sharing, spatial correlation down sampling, etc., which not only greatly reduces the training parameters of the network, but also makes the network structure simple and adaptable. The hierarchical structure of the convolutional neural network is generally: input layer → convolution layer → pooling layer → (repeating: convolution layer → pooling layer) → full connection layer → output layer. Among them, the input layer is mainly to preprocess the original image. The convolutional layer is an important part of the convolutional neural network, which consists of a series of filter banks. In the convolutional layer, the weight of each neuron connected data window is fixed, and each neuron focuses on only one characteristic. Neurons are filters in image processing. Each neuron in a convolutional layer focuses on an image feature, such as vertical edges, colors, textures, etc. All neurons are extractors that extract features from the entire image. The pooling layer is mainly to reduce the spatial size of the data, thereby reducing the number of parameters to be learned in the neural network, reducing the resource consumption of the computer, and effectively preventing the over-fitting problem in the training process. The fully connected layer is used to extract the features of the previously learned full graph and convert them into a classifier to serve as a classification[5-6].

2.2 Full Convolutional Network

U-net is a semantic segmentation network based on a full convolutional network. The structure of the entire network is like a "U" type, so it is called U-Net. U-net uses an encoder-decoder structure in which the encoder gradually reduces the spatial dimensions of the pooling layer, and the decoder gradually repairs the details and spatial dimensions of the target object. Normally, there is a quick connection between the encoder and the decoder to help the decoder better repair the details of the target object. The network does not have a fully connected layer, only convolution and down sampling, which is an end-to-end approach where the input is an image and the output is an image[9-11].

2.3 U-net Network

U-net is a semantic segmentation network based on a full convolutional network. The structure of the entire network is like a "U" type, so it is called U-Net. U-net uses an encoder-decoder structure in which the encoder gradually reduces the spatial dimensions of the pooling layer, and the decoder gradually repairs the details and spatial dimensions of the target object. Normally, there is a quick connection between the encoder and the decoder to help the decoder better repair the details of the target object. The network does not have a fully connected layer, only convolution and down sampling, which is an end-to-end approach where the input is an image and the output is an image[9-11].

3 Data Description and Experiment

Based on the U-net network architecture, programming is implemented on Pycharm using the Python language. The experimental steps are as follows: (1) First, the multi-category tag data is converted into binary tag data containing only buildings and backgrounds. (2) The four large remote sensing image cut at random, i.e., randomly generated x, y coordinates, then a small image of 256*256 pixels at this coordinate is obtained, followed by data expansion processing. A total of 100,000 images were obtained, and a training set and a verification set were generated in a 4:1 ratio. (3) Using the U-net network for training, the input is 100000 remote sensing images, and the training round is 10 rounds. The weights obtained in each round of training are saved during the training, and the optimal model is obtained. After the training, another large-size remote sensing image was divided into several 256*256 small images as test sets, and the models saved in each round were tested, and finally get 10 to extract a good picture. Accurately analyze the segmented image obtained from the test and the original remote sensing image (true value), and then obtain the final result.

3.1 Data Preparation

The data set used in this experiment is the data provided by CCF Big Data and Computer Intelligence Competition (BDCI), which is a high-resolution remote sensing image of a city in southern China in 2015, including the surface overlay sample image is visually interpreted by the remote sensing image. The spatial resolution of the image is sub-meter, the spectrum is visible light (R, G, B), and the coordinate information has been removed. The dataset contains 5 RGB remote sensing images with label (size range: 3000×3000~6000×6000). The samples are polygons that are visually interpreted and manually sketched. The samples provided are simplified into five categories: Vegetation (mark 1), Building (mark 2), Water (mark 3), Road (mark 4), and others (mark 0). Among them, farmland, woodland, grassland are classified as vegetation class. The time span of image collection is from April to August, and the surface changes are relatively large. Some farmland and woodland are in the state of harvesting or felling, so they are all classified as vegetation. Since this experiment is a two-category, building and non-building (background), only two types of tags are needed. Sample data containing five types of tags is first processed and converted into sample data containing two types of tags. This can be achieved by introducing the opencv library in a python environment. The final training images are 1.png, 2.png, 3.png, 4.png, and the corresponding annotation images are label1.png, label2.png, label3.png, and label4.png. The experimentally verified image is 5.png and the corresponding labeled image label5.png.

3.2 Data Set Preparation

After the completion of the work, we got five image contains two types of labels, but these images can not be directly used for network training, on the one hand can not afford the computer's memory, and the size of different images are also different. On the other hand, performing network training requires a large amount of images. The experimental images are divided into training set, verification set and test set. In this stage, the data set of 100000 pieces is needed, and is divided into training set and verification set by 4:1 in the training process. Therefore, firstly, the images with the numbers 1~4 are randomly cut, that is, the x,y coordinates are randomly generated. By Gamma conversion, random rotation, adding noise, the four image smoothing operation for data expansion purposes, the size of the finally obtained data sets 100,000[12-14].

3.3 Network Training

Once the data set is ready, you can start network training. The training is mainly divided into five parts, and the following is a step-by-step description of the experimental content.

Read in the Data Set. First, use OpenCV's imread definition to load the image function, then specify the size of the validation set to account for 20% of the total, read the data set, use the listdir function to list the directories and files under src_1, and use the shuffle function to randomly confuse the order of the images. , divided into training set and verification set according to the ratio of 4:1. Where train_set is the training set, val_set is the validation set, and train_url is the total data set.

Model reconstruction. Define a generator function for reading the training set data and use the yield statement to return each result. The original image of the training set and the corresponding tag image are respectively loaded and converted into an array, which are sequentially arranged.

Read Validation Set Data. As above, define a generator function for reading validation set data, using the yield statement to return each result. The original image of the verification set and the corresponding tag image are respectively loaded and converted into an array, which are sequentially arranged. It should be noted that during the training process, the verification set does not participate in the training of the model, that is, the generation of weights. The verification of the model accuracy is performed only after the weight generation, and the process is implemented in the training part[15-16].

Define the U-net Model. The U-net model consists of 28 layers, including 19 convolutional layers, 1 input layer, 4 pooling layers, and 4 up sampling layers. The connection method is: input layer → convolution layer - convolution layer - pooling layer (4 times)

→ convolution layer - convolution layer → up sampling layer - convolution layer - convolution layer (4 times) → convolution Layer (output layer). The model is shown in Figure 1.

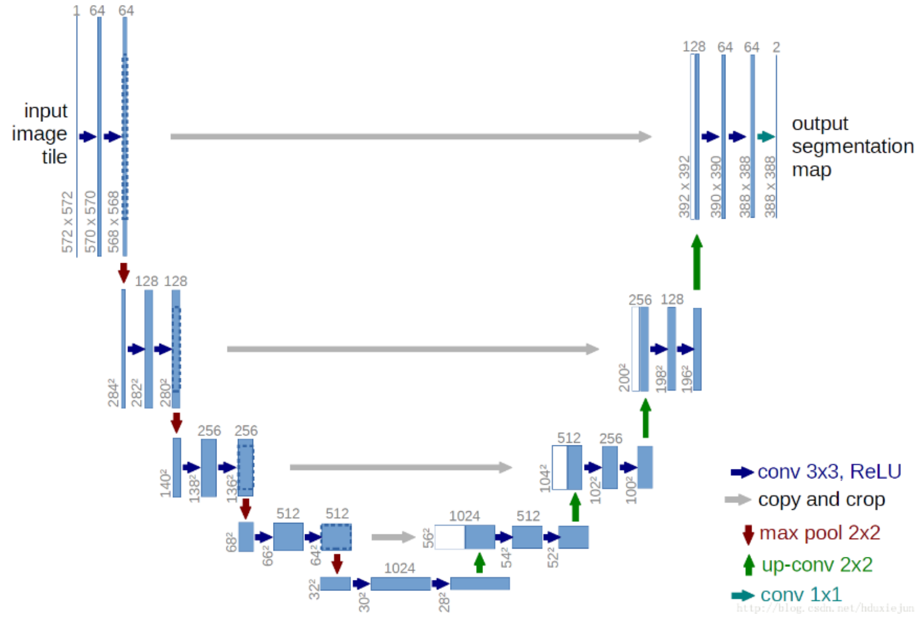

Fig. 1. U-net network architecture

Use conv1 as an example to illustrate the use of convolutional layer parameters. The model is all based on Conv2D (two-dimensional convolutional layer), which is a spatial convolution of the image. The first parameter is the number of convolution kernels used by this layer (i.e., the latitude of the output); the second parameter (3, 3) is the width and length of the convolution kernel; the activation function used is ReLU, which maps the K-dimensional real field to the (0, inf) interval. For details, refer to the previous introduction to the activation function, the padding parameter refers to the complement "0" strategy of the window during the sliding process, and "same" represents the convolution result at the reserved boundary, which makes the output shape the same as the input shape. In addition, the input accepted by this layer is the output of the previous layer.

Take pool1 as an example to illustrate the use of pooling layer parameters. The model uses MaxPooling2D, which is to apply the maximum pooling to the airspace signal. Pool_size represents the down sampling factor in both directions (vertical, horizontal), taking (2, 2) will make the picture half of the original length in both dimensions. In addition, the input accepted by this layer is the output of the previous layer.

The up6 is used as an example to illustrate the use of the up sampling layer parameters. The model uses UpSampling2D, which is to repeat the size [0] and size [1] times of the row and column of the data. Size is the row and column up sampling factor, and other unlisted parameters use the default value. Use the concatenate function to join the two convolutional layer matrices on the specified axis.

Conv10 is the output layer, using the (1, 1) convolution kernel instead of the common fully connected output layer. The activation function used is sigmoid because it deals with the normalized binary classification problem. The K-dimensional real number field can be compressed to approximately 0, 1 binary values. Finally, the optimization method chosen by U-net is Adam. Adam is a first-order optimization algorithm that can replace the traditional stochastic gradient descent process. It can iteratively update the weight of the neural network based on the training data. The learning rate lr of the model is set to 0.01. The loss function is the binary entropy loss (binary_crossentropy), and the evaluation method is two-category accuracy (binary_accuracy).

Cycle Training. Once the data and model are ready, you can start training. The whole process of training is carried out on the GPU. The number of training EPOCHS is 10 times, each time is about 50 minutes, the total training time is about 10 hours, and the image BS is 16 samples per sample. In the training, the loss is first calculated by forward calculation, then the gradient on each BS is calculated by back propagation. Finally, the gradient parameters are used to update the weight parameters. The weight model obtained in each round is automatically saved during the training. In addition, the remaining time, the number of remaining data, and the loss value and the binary_accuracy value of the training set are displayed during the training. After the training, the loss value and the binary_accuracy value on the test set and the verification set of this round are displayed.

3.4 Expected Outcome

By recording the accuracy of the training set and the verification set during the training process, that is, both are above 90%, it can be predicted that the accuracy of the test set will be between 65% and 85%. Considering the accuracy of the results that have been studied in this field, the accuracy of this experiment is 65%~75%. If the experimental results are within this interval, it is feasible to apply the U-net model to remote sensing image information extraction.

4 Result Analysis

After the training of the network model is completed, the feasibility of the model needs to be tested. The test data is the fifth remote sensing image and its corresponding label image. The specific contents are as follows: (1) The ten models obtained are tested separately. (2) Analysis using a confusion matrix. (3) Analyze the loss curve and accuracy curve of the model.

4.1 Set Test

As with the training set and the validation set, it is also necessary to create a test set when testing, rather than feeding the original image directly into the model. The difference is that the test set uses a fixed step size to slice the original image. The image size is 256*256. After the test set is ready, the U-net model and weights are loaded, and then the test set image is sent to the model for testing, and finally a sample of the extracted building is obtained. Because there are ten weights, the program is run ten times repeatedly, and finally ten images are obtained.

4.2 Confusion Matrix Analysis

The confusion matrix is a standard format for the accuracy evaluation, which is in the form of a matrix of n rows and n columns. In the confusion matrix, each column represents the forecast category of the data, the total number of each column represents the total number of data predicted for that category, each row represents the real category of the data, the total number of data for each row represents the total number of such data, the number of each column indicates that the true value is predicted as the number of that class. As shown in Table 1, the sum of the first row is A+B, indicating that there are A+B samples. The first row indicates that Class A has the correct classification and B is classified into Class 2; the second line indicates that D is classified correctly in class 2, and C is divided into class 1.

Table 1. Example of confusion matrix

——		Forecast	
		Class 1	Class 2
Real	Class 1	A	B
	Class 2	C	D

After comparison, the better test images from the ten images are compared with the known images as shown in Figure 2 and Figure 3.

The two images are compared and analyzed by the confusion matrix, and the results of the confusion matrix analysis are shown in table 2:

According to Table 2, the total number of non-building pixels is 59,093,612, of which 41,317,449 are paired, and the total number of pixels in the building is 2,500,624, of which 177,1975 are pairwise, with an accuracy rate of 70.86%.

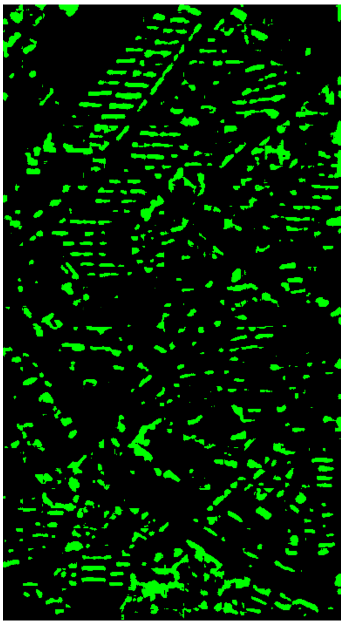

Fig. 2. Forecast Image

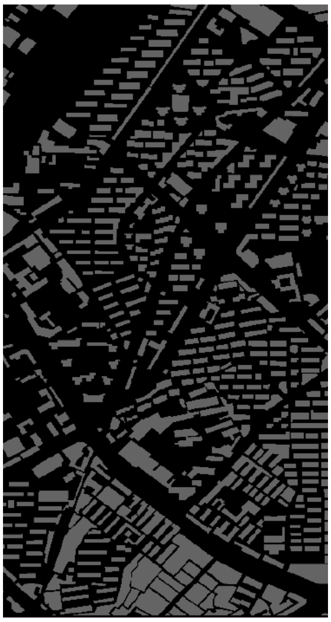

Fig. 3. Real Image

Table 2. Confusion matrix analysis results

——		Forecast	
		building	Non-building
Class	Non-building	41317449	17776163
	building	728649	1771975

4.3 Loss Curve and Accuracy Curve Analysis

Through the tensorboard visualization tool tensorboard and the loss values and precision values saved during the training process, the tensorboard can directly generate loss curves and precision curves[17-18]. Figure 4 and figure 5 show the loss and accuracy values of the training set, respectively.

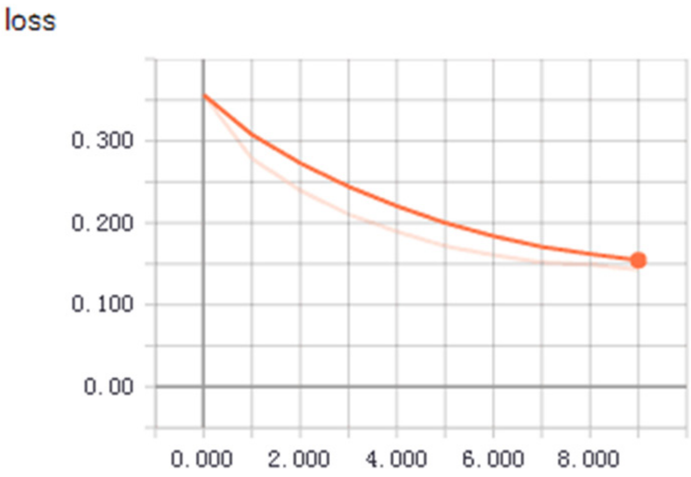

Fig. 4. Training set loss function curve

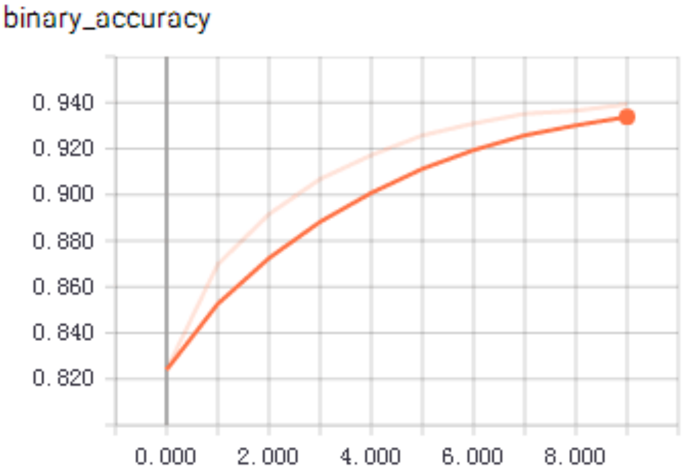

Fig. 5. Training set accuracy function curve

It is not difficult to see from the curve in figure 4 that the loss on the training set is gradually reduced, and the speed of convergence is moderate, but eventually it does not reach the stable value, and may be further reduced. As can be seen from figure 5, the accuracy of the training model is steadily increasing, gradually approaching 0.940.

The experiment also obtained the loss curve and accuracy curve on the verification set. Figure 6 and figure 7 show the loss and accuracy values of the verification set, respectively.

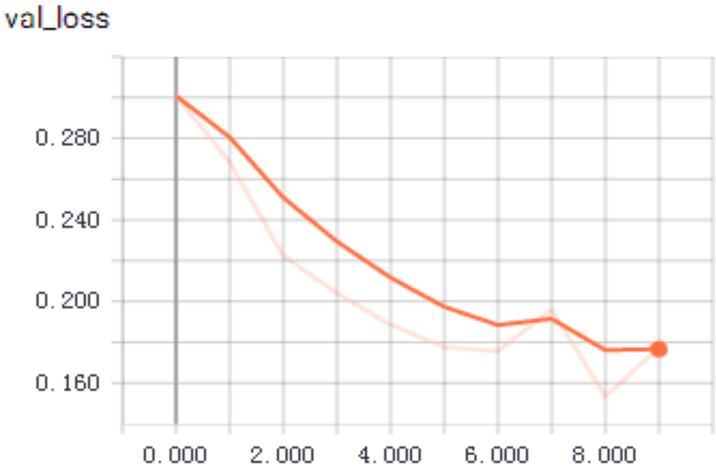

Fig. 6. Verification set loss function curve

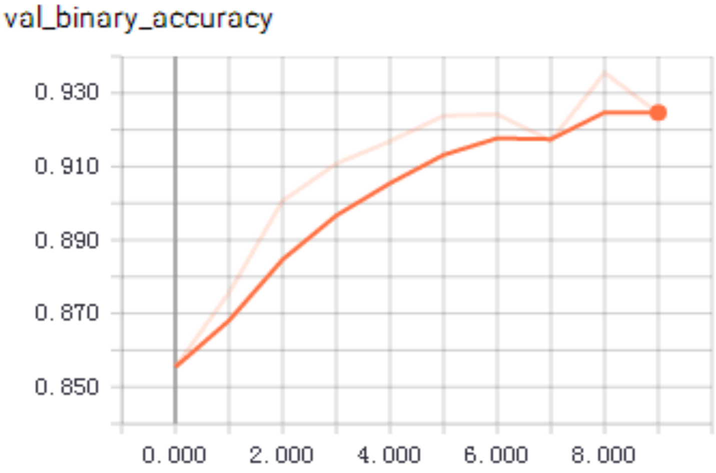

Fig. 7. Verify set accuracy function curve

It can be seen from Figure 4.6 that the loss on the verification set of the first six rounds has been decreasing, and the seventh round has produced a local peak, and then continues to decline. The first six rounds of accuracy of the corresponding verification set have been rising, the sixth round reached 0.918, the seventh round suddenly dropped, and then rose to about 0.925. It is not difficult to find that as the learning ability of the model continues to increase, it leads to over-fitting after the seventh round, which makes the precision increase rapidly. The first six rounds of the actual model accuracy rise curve are reasonable.

5 Conclusions

Through the above experiments, combined with data analysis, the following conclusions are drawn on the building information extraction method based on U-net network remote sensing image: (1) Compared with the manual extraction method, the end-to-end training method significantly improves the extraction efficiency of remote sensing image information.(2) Allowing the computer to automatically extract image features effectively reduces the one-sidedness and inefficiency of the artificial design features.(3) It is feasible to apply U-net network to large-scale, high spatial resolution remote sensing image information extraction, which provides new ideas and methods for remote sensing image information extraction. It provides new ideas and new methods for remote sensing image information extraction, which provides more abundant technical support for the application of high resolution remote sensing satellites in China.

6 Acknowledgment

This work was supported by National Key Research and Development Project of China [Grant Number 2016YFC0801602].

7 References

[1] L.Y. Feng. Research on Land Use Information Extraction from High Resolution Remote Sensing Image Based on Deep Learning[D]. Zhejiang University, 2017.

[2] J.B. Sun. Remote sensing principle and application [M].Wuhan:Wuhan Uni-versity Press,2013,127-230.

[3] G.E. Hinton and R.R. Salakhutdinov. Reducing the dimensionality of data with neural networks[J]. Science,313(5786):504-507,2006. https://doi.org/10.1126/science.1127647

[4] Karen Simonyan，Andrew Zisserman.Very deep convolutional networks for large-scale image recognition[J]. arXiv：1409.1556v6，2015.

[5] Jonathan Long，Evan Shelhamer，Trevor Darrell.Fully Convolutional Net-works for Semantic Segmentation[J]. arXiv：1411.4038v2，2015.

[6] Jifeng Dai，Kaiming He，Yi Li，Shaoqing Ren，Jian Sun.Instance-sensitive Fully Convolutional Networks[J]. arXiv：1603.08678v1，2016.

[7] G.E. Hinton and R.R. Salakhutdinov. Reducing the dimensionality of data with neural networks[J]. Science,313(5786):504-507,2006. https://doi.org/10.1126/science.1127647

[8] Jimmy Lei Ba，Jamie Ryan Kiros，G.E. Hinton.Layer Normalization[J]. arXiv：1607.06450v1，2016.

[9] G. Wang and Jinyong Chen et al. Research on Remote Sensing Image Infrastruc-ture Target Detection Based on Deep Learning[J]. Radio Engineering, 2018,48(3):219-224.

[10] S.C. Li. Image Ink Style Rendering Application Based on Deep Learning [D]. Nanjing University,2017.

[11] Z. Wang. Research on Fast Target Detection Technology Based on Deep Learn ing [D]. Tianjin University of Technology,2017.

[12] Z.X. Zhao. Research on License Plate Recognition Technology Based on Deep Learning [D]. Qingdao University of Science and Technology,2017.

[13] Q.P. Bao. Classification and retrieval of clothing images based on deep learn-ing [D]. Zhejiang University,2017.
[14] Diederik P.Kingma，Jimmy Lei Ba. Adam_ A method for stochastic optimiza-tion[J]. arXiv：1412.6980v9，2017.
[15] Dawei Wen,Xin Huang,Hui Liu,et al.Semantic Classification of Urban Trees Using Very High Resolution Statellite Imagery[J].IEEE Journal of Se-lected Topics in Applied Earth Observation & Remote Sensing,2017,10(4):1413-1424.
[16] Jx Wang，Z Kurth-Nelson，D Tirumala，H Soyer，JZ Leibo，R Munos，et al.Learning to Reinforcement Learn [J]. arXiv：1611.05763v3，2017.
[17] Ian Goodfellow，Yoshua Bengio，Aaron Courville.Deep Learn-ing[M].American：The MIT Press，2016，167-371.
[18] Jan Erik Solem. Python computer vision programming [M]. Beijing: People's Posts and Telecommunications Press,2014,25-342.

8 Authors

Jingtan Li is currently an under graduate student of University of Science and Technology Liaoning, Anshan, China, 114051 (ijt19950817110@163.com).

Maolin Xu is currently a professor of University of Science and Technology Liaoning, Anshan, China, 114051, His research interest includes mine monitoring and measurement data processing (xml1964@163.com).

Hongling Xiu is currently a master graduate student of University of Science and Technology Liaoning, Anshan, China, 114051, Her research interest includes remote sensing image processing and application (1063539137@qq.com).

Article submitted 03 August 2018. Resubmitted 29 October 2018. Final acceptance 05 September 2018. Final version published as submitted by the authors.