IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 30, 2019, accepted May 14, 2019, date of publication May 20, 2019, date of current version June 3, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2917952

DenseU-Net-Based Semantic Segmentation of Small Objects in Urban Remote Sensing Images

RONGSHENG DONG, XIAOQUAN PAN, AND FENGYING LI
Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China
Corresponding author: Fengying Li (lfy@guet.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61762024, and in part by the Natural Science Foundation of Guangxi Province under Grant 2017GXNSFDA198050 and Grant 2016GXNSFAA380054.

ABSTRACT Class imbalance is a serious problem that plagues the semantic segmentation task in urban remote sensing images. Since large object classes dominate the segmentation task, small object classes are usually suppressed, so the solutions based on optimizing the overall accuracy are often unsatisfactory. In the light of the class imbalance of the semantic segmentation in urban remote sensing images, we developed the concept of the Down-sampling Block (DownBlock) for obtaining context information and the Up-sampling Block (UpBlock) for restoring the original resolution. We proposed an end-to-end deep convolutional neural network (DenseU-Net) architecture for pixel-wise urban remote sensing image segmentation. The main idea of the DenseU-Net is to connect convolutional neural network features through cascade operations and use its symmetrical structure to fuse the detail features in shallow layers and the abstract semantic features in deep layers. A focal loss function weighted by the median frequency balancing (MFB_Focal_{loss}) is proposed; the accuracy of the small object classes and the overall accuracy are improved effectively with our approach. Our experiments were based on the 2016 ISPRS Vaihingen 2D semantic labeling dataset and demonstrated the following outcomes. In the case where boundary pixels were considered (GT), MFB_Focal_{loss} achieved a good overall segmentation performance using the same U-Net model, and the F1-score of the small object class ''car'' was improved by 9.28% compared with the cross-entropy loss function. Using the same MFB_Focal_{loss} loss function, the overall accuracy of the DenseU-Net was better than that of U-Net, where the F1-score of the ''car'' class was 6.71% higher. Finally, without any post-processing, the DenseU-Net+MFB_Focal$_{\text{loss}}$ achieved the overall accuracy of 85.63%, and the F1-score of the ''car'' class was 83.23%, which is superior to HSN+OI+WBP both numerically and visually.

INDEX TERMS Class imbalance, deep convolutional neural networks, median frequency balancing, semantic segmentation, urban remote sensing images.

I. INTRODUCTION

The semantic segmentation task of remote sensing images classifies each pixel in remote sensing images. When dealing with very-high spatial resolution (VHR) remote sensing images, the spectral resolution and spatial resolution are mutually constrained [1], so the sensor makes a trade off with the spectral resolution to gain a high spatial resolution. Therefore, when analyzing the spectral information of single pixels, the spatial context needs to consider the spatial features, such as those that are textural [2] or morphological [3]. These features consider the neighborhood around a pixel as part of its own features and allow for the placement of spectral features in context.

The above spatial features need manual extraction, and the aim of deep learning is to train a parametric system for feature extraction jointly with classifiers in an end-to-end manner to avoid the manual extraction of the spatial feature. With the development of deep learning, the Convolutional Neural Networks (CNN) has been pioneered by Krizhevsky *et al.* [4]. Researchers have also made a series of breakthroughs in computer vision, such as image classification, object detection, semantic segmentation, and other tasks. The great success of CNNs is mainly due to their excellent feature description for visual data. Deep networks extract features better than artificial feature engineering [5]. The development of hardware

The associate editor coordinating the review of this manuscript and approving it for publication was Shagufta Henna.

technology and the increasing number of pre-training models have increased the influence of CNNs in the field of remote sensing research.

Remote sensing images are often characterized by complex data properties in the form of heterogeneity and class imbalances, as well as overlapping class-conditional distributions [6]. Class imbalance is a public issue of the semantic segmentation task of urban remote sensing images [7]. The classes of land cover in remote sensing images are often highly imbalanced, and some land cover classes are relatively small, which is a challenge to the semantic segmentation task. Since the small object classes are usually suppressed, the solutions based on optimizing the overall accuracy are often unsatisfactory [8].

II. RELATED WORK

In 2012, the classic deep CNN, AlexNet [4], was developed and achieved the best results of the year in the image classification task of ImageNet image recognition competition. Subsequently, the deeper ZFNet [9], VGGNet [10], and GoogLeNet [11] were successively proposed.

These CNNs are quite effective for extracting deep features and classifying scene images, but do not have much effect on the pixel-level semantic segmentation tasks. In 2015, Long *et al.* [12] proposed that full convolutional network (FCN) achieved the best results in the image segmentation task of the PASCAL VOC visual recognition competition. The FCN consists of an encoder and decoder. The encoder is similar to the traditional CNN in that it extracts deep abstract features, and the decoder restores these features to dense prediction maps of the same size as the input images. In the same year, Ronneberger *et al.* [13] designed a multi-scale U-Net based on FCN. The architecture not only achieves the consistency of the input and output image resolution, but also fuses shallow and deep features; U-Net was successfully applied to biomedical images. U-Net achieved the best results of segmentation at that time and was applied to medical images and natural scene images.

In recent years, deep learning has been applied to the semantic segmentation of remote sensing images. In 2012, Mnih and Hinton [14] used deep neural networks for semantic segmentation in aerial images, two kinds of loss functions were proposed to reduce the influence of omission noise and registration noise on classifiers. This method achieved better results than traditional methods and showed the potential of applying deep learning technology for semantic segmentation in remote sensing images. In 2013, Mnih [15] assembled two datasets released by the state of Massachusetts and a dataset released by the state of New York, the target maps for all three datasets were generated using data from the OpenStreetMap project, and published the first public dataset of large roads and buildings detection. It further promoted the development of semantic segmentation in remote sensing images. In 2016, Kampffmeyer *et al.* [6] adopted 2 different structures: the first was a patch-based pixel classification that used 65 × 65 pixel blocks for dense segmentation, and the second was pixel-to-pixel segmentation, where the convolutional layers of the contracting path are followed by a deconvolutional layer, and the features are directly upsampled back to the original image resolution. The disadvantage of the second approach is that some details of the images are lost. In the same year, Saito *et al.* [16] proposed a new channel-wise inhibited softmax (CIS) function instead of the original softmax function, which effectively solved the multi-object semantic segmentation in remote sensing images. In 2017, Volpi and Tuia [17] proposed a CNN-FPL model based on U-Net for generating dense labeling maps, which had the advantage of the convolutional layers of the contracting path being followed by multi-layer deconvolutional layers to restore the features back to the original image resolution layer by layer. This approach helps to retain the details of the images. In the same year, Iglovikov *et al.* [18] applied the idea of a skip connection to the U-Net and concatenated the features of the shallow and deep layers to improve the utilization of the feature information. Liu *et al.* [19] proposed an Hourglass-Shape Networks (HSN) and introduced the Inception module to provide the network with multi-scale receptive areas with rich context. In 2018, Gao *et al.* [20] proposed a multiple feature pyramid network (MFPN), which is used for road extraction of remote sensing images, and given a weighted balance loss function is presented to solve the class imbalance problem caused by the sparseness of roads.

We propose an end-to-end fully convolutional network, DenseU-Net, which is based on the work of U-Net [13]. DenseU-Net connects the CNN features through cascade operations and through its symmetrical structure composed of continuous DownBlocks and UpBlocks. The features in the shallow layers, such as color and texture, are merged with the abstract semantic features in the deep layers through the skip connection. This approach alleviates the problem with the accuracy, which degrades as the network depth increases. MFB_Focal_{loss}, which is a focal loss weighted by the median frequency balancing, is given. MFB_Focal_{loss} improves the overall segmentation accuracy, especially the segmentation accuracy of small object classes. The main contributions of this paper are as follows:

1) A focal loss function weighted by the median frequency balancing (MFB_Focal_{loss}) is given. The loss of the classes is weighted by the ratio of the median class frequency in the training set and the actual class frequency, and a factor is introduced based on the standard cross entropy loss function to reduce the relative loss for well-classified examples, putting more focus on hard-misclassified examples. In the case of using the same U-Net, the overall accuracy of MFB_Focal_{loss} is better than CE_{loss}. In particular, the F1-score of the ''car'' class increases by 9.28%.
2) Based on the U-Net structure and the idea of dense connection, the concept of the down-sampling block (DownBlock) for obtaining context information and the up-sampling block (UpBlock) for restoring the

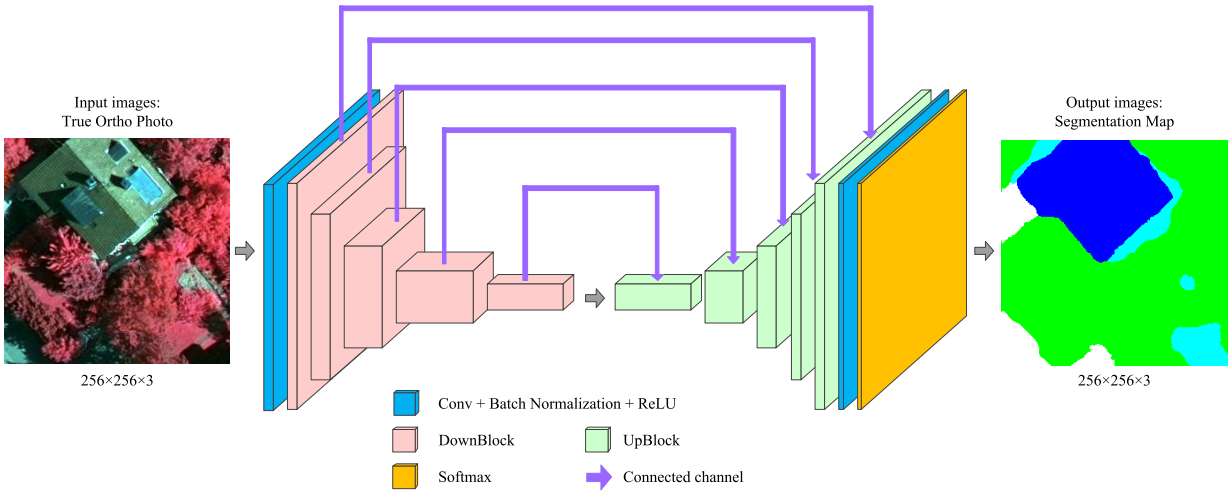


FIGURE 1. DenseU-Net network architecture.

original resolution are given, and an end-to-end deep convolutional neural network (DenseU-Net) architecture is proposed. In the case of using the same MFB_Focal_{loss}, the F1-score of each class, average F1-score, and overall accuracy of DenseU-Net are better than those of U-Net. In particular, the F1-score of the "car" class is 6.71% higher.

3) Without any post-processing, DenseU-Net+MFB_Focal_{loss} achieves an overall accuracy of 85.63%, which is still better than the 85.39% of HSN+OI+WBP. The F1-score of the "car" class for DenseU-Net+MFB_Focal_{loss} exceeds 7.28%.

III. PROPOSED METHODS

DenseU-Net consists of a contracting path for obtaining context information and an expanding path for precise positioning, as shown in Figure 1. The contracting path of DenseU-Net is composed of 5 consecutive downsampling blocks (DownBlocks), and the expanding path is also composed of 5 consecutive upsampling blocks (UpBlocks). Each DownBlock is connected to the corresponding UpBlock from the expanding path by a connected channel. After each DownBlock, the number of feature dimensions doubles, and after each UpBlock, the number of feature dimensions is halved. The network does not use any fully connected layers, and the Softmax layer is used for the dense prediction of the features of the output. The network input images consist of 3 channels of true ortho photo images (TOP images). The detailed parameters of each layer are shown in Table 1.

A. DOWNBLOCK

The DownBlock structure was inspired by Dense connections [21], and the structure is shown in Figure 2. The input x of the DownBlock are feature maps of the D-dimensional $H \times W$. The input feature maps for feature extraction occur through 2 densely connected convolutional layers. The 2 convolutional layers both use D-dimensional convolution kernels with a filter size of 3×3 and stride size of 1×1. The output y_2 of the second convolutional layer, input x of DownBlock, and output y_1 of the first convolutional layer are connected by a cascade operation. The connected $3D$-dimensional feature

TABLE 1. Detailed parameters of each layer of DenseU-Net.

Layer Name		Kernel Number	Kernel Size
Inconv		64	3×3
DownBlock0	Conv0_1	64	3×3
	Conv0_2	64	3×3
	Conv0_3	64	1×1
	Maxpool0	64	2×2
DownBlock1	Conv1_1	128	3×3
	Conv1_2	128	3×3
	Conv1_3	128	1×1
	Maxpool1	128	2×2
DownBlock2	Conv2_1	256	3×3
	Conv2_2	256	3×3
	Conv2_3	256	1×1
	Maxpool2	256	2×2
DownBlock3	Conv3_1	512	3×3
	Conv3_2	512	3×3
	Conv3_3	512	1×1
	Maxpool3	512	2×2
DownBlock4	Conv4_1	512	3×3
	Conv4_2	512	3×3
	Conv4_3	512	1×1
	Maxpool4	512	2×2
UpBlock0	TransposedConv0	512	2×2
	Conv5_1	512	1×1
	Conv5_2	512	3×3
	Conv5_3	512	3×3
	Conv5_4	512	1×1
UpBlock1	TransposedConv1	512	2×2
	Conv6_1	256	1×1
	Conv6_2	256	3×3
	Conv6_3	256	3×3
	Conv6_4	256	1×1
UpBlock2	TransposedConv2	256	2×2
	Conv7_1	128	1×1
	Conv7_2	128	3×3
	Conv7_3	128	3×3
	Conv7_4	128	1×1
UpBlock3	TransposedConv3	128	2×2
	Conv8_1	64	1×1
	Conv8_2	64	3×3
	Conv8_3	64	3×3
	Conv8_4	64	1×1
UpBlock4	TransposedConv4	64	2×2
	Conv9_1	64	1×1
	Conv9_2	64	3×3
	Conv9_3	64	3×3
	Conv9_4	64	1×1
Outconv		6	3×3

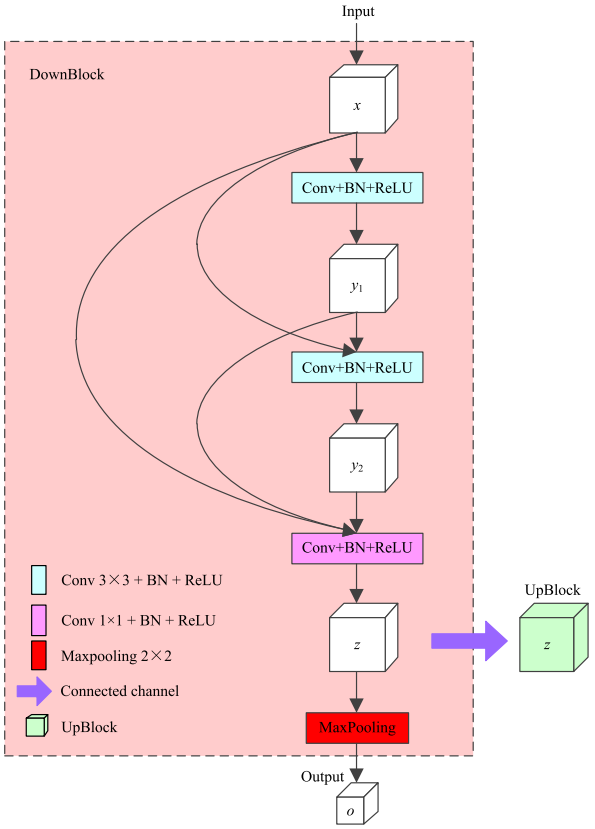


FIGURE 2. Details of the DownBlock.

maps are reduced to D-dimensions by the D-dimensional convolution kernels with a filter size of 1×1. There is a Rectified Linear Unit (ReLU) layer and a Batch Normalization (BN) layer that come after the convolutional layers. The reduced dimensional features not only serve as the input of the max pooling layer, but also pass the reduced dimensional features to the corresponding UpBlock.

The formal definition of the DownBlock structure is as follows:

DownBlock $= < x, W_i, \sigma(\cdot), y_i, cas, W_d, z, mp, o >$

The structure is defined as a tuple with the following components:

x: x indicates the input of the DownBlock;

W_i: W_i indicates the i-th 3×3 convolution operation, $x = 1, 2$;

$\sigma(\cdot)$: $\sigma(\cdot)$ indicates a composite function, i.e., Rectified Linear Unit and Batch Normalization;

y_i: y_i indicates the output of the i-th 3×3 convolution operation, $i = 1, 2$;

cas: cas indicates the cascade operation;

W_d: W_d indicates the 1×1 convolutional dimension reduction operation;

z: z indicates the reduced dimensional features;

mp: mp indicates the max pooling operation; and

o: o indicates the output of the DownBlock.

Suppose x represents the D-dimensional input features of the DownBlock; the outputs y_1 of the first convolutional layer are

$$y_1 = \sigma(W_1 \cdot x), \tag{1}$$

where W_1 indicates the first convolution operation and $\sigma(\cdot)$ indicates a composite function, i.e., the Rectified Linear Unit and Batch Normalization.

The outputs y_1 of the first convolutional layer are connected to the inputs x of the DownBlock by the cascade operation. The connected $2D$-dimensional feature maps serve as the inputs of the second convolutional layer, and the outputs y_2 of the second convolutional layer are

$$\begin{aligned} y_2 &= \sigma(W_2(\sigma(W_1 \cdot x) + x)) \\ &= \sigma(W_2(y_1 + x)), \end{aligned} \tag{2}$$

where W_2 indicates the second convolution operation.

Similarly, the outputs y_1 of the first convolutional layer, outputs y_2 of the second convolutional layer, and inputs x of the DownBlock are connected by the cascade operation. The connected $3D$-dimensional feature maps are reduced to D-dimensions by the D-dimensional convolution kernels with a filter size of 1×1. This approach is beneficial for improving the computational efficiency. The reduced dimensional features z are

$$\begin{aligned} z &= W_d(\sigma(W_2(\sigma(W_1 \cdot x) + x)) + \sigma(W_1 \cdot x) + x) \\ &= W_d(y_2 + y_1 + x). \end{aligned} \tag{3}$$

Finally, the reduced dimensional features z not only serve as the input of the max pooling layer, but also pass the reduced dimensional features to the corresponding UpBlock.

B. UPBLOCK

UpBlock helps to integrate more accurate output, and its structure is similar to the DownBlock, as shown in Figure 3. The input x_1 of the UpBlock structure contains the feature maps of D-dimensional $H \times W$. The input feature maps are upsampled by transposed convolution with a stride size of 2×2, and the size of the feature maps is restored to $2H \times 2W$. Then, the upsampled features are fused with the same-sized feature maps of the corresponding DownBlock from the contracting path. The fused feature maps are reduced to D-dimensions by the D-dimensional convolution kernels with a filter size of 1×1. The reduced dimensional features for feature extraction through 2 densely connected convolutional layers occur through 2 convolutional layers that both use D-dimensional convolution kernels with a filter size of 3×3 and stride size of 1×1. The output features y_4 of the second convolutional layer, input y_2 of the 2 densely connected convolutional layers, and output y_3 of the first convolutional layer are connected by the cascade operation. The connected $3D$-dimensional feature maps are reduced to D-dimensions by the D-dimensional convolution kernels with a filter size of 1×1. There is a Rectified Linear Unit (ReLU) layer and a Batch Normalization (BN) layer that come after the convolutional layers.

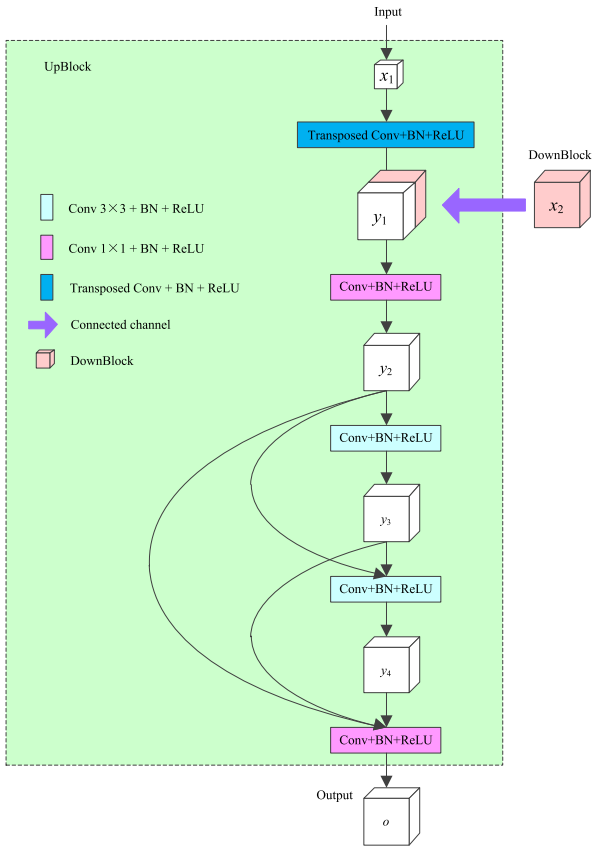


FIGURE 3. Details of the UpBlock.

The formal definition of the UpBlock structure is as follows:

UpBlock = $< x_1, W_t^T, \sigma(\cdot), x_2, W_i, y_i, cas, W_d, o >$

The structure is defined as a tuple with the following components:

x_1: x_1 indicates the input of the UpBlock;

W_t^T: W_t^T indicates the transposed convolution operation;

$\sigma(\cdot)$: $\sigma(\cdot)$ indicates a composite function, i.e., the Rectified Linear Unit and Batch Normalization;

x_2: x_2 indicates the feature transmitted by the corresponding DownBlock;

W_i: W_i indicates the i-th 3×3 convolution operation, $i = 1, 2$;

y_i: y_i indicates the output of the i-th convolution operation, $i = 1, 2, 3, 4$;

cas: cas indicates the cascade operation;

W_d: W_d indicates the 1×1 convolutional dimension reduction operation; and

o: o indicates the output of the UpBlock.

Suppose x_1 represents the D-dimensional input features of the UpBlock, and the outputs y_1 of the transposed convolutional layer are

$$y_1 = \sigma(W_t^T \cdot x_1), \tag{4}$$

where W_t^T indicates the transposed convolution operation and $\sigma(\cdot)$ indicates a composite function, i.e., the Rectified Linear Unit and Batch Normalization.

The outputs y_1 of the transposed convolutional layer are cascaded to the features x_2 transmitted by the corresponding DownBlock. The cascaded feature maps are reduced to D-dimensions by the D-dimensional convolution kernels with a filter size of 1×1. The reduced dimensional features y_2 are

$$\begin{aligned} y_2 &= W_d(\sigma(W_t^T \cdot x_1) + x_2) \\ &= W_d(y_1 + x_2). \end{aligned} \tag{5}$$

The reduced dimensional features y_2 are used as the inputs of the 2 densely connected convolutional layers, and the outputs y_3 of the first convolutional layers are

$$y_3 = \sigma(W_1 \cdot y_2), \tag{6}$$

where W_1 indicates the first convolution operation.

The outputs y_3 of the first convolutional layer are connected to the inputs y_2 of the 2 densely connected convolutional layers by the cascade operation. The connected $2D$-dimensional feature maps serve as the input of the second convolutional layer. The outputs y_4 of the second convolutional layer are

$$y_4 = \sigma(W_2(y_3 + y_2)), \tag{7}$$

where W_2 indicates the second convolution operation.

Finally, the outputs y_3 of the first convolutional layer, outputs y_4 of the second convolutional layer and inputs y_2 of the 2 densely connected convolutional layers are connected by the cascade operation. The connected feature maps are reduced to D-dimensions by the D-dimensional convolution kernels with a filter size of 1×1. The outputs o of the UpBlock are

$$o = W_d(y_4 + y_3 + y_2). \tag{8}$$

C. LOSS FUNCTION

Due to the skewed distribution of the ground objects, remote sensing samples encounter the class imbalance problem. The class imbalance problem typically occurs when, in a classification problem, there are many more instances of some classes than others. In such cases, the standard classifiers tend to be overwhelmed by the larger classes and ignore the smaller ones [22]. Class imbalance causes 2 problems: (1) training is inefficient, as most locations are easy negatives that contribute no useful learning signals; (2) en masse, the easy negatives can overwhelm training and lead to degenerate models [23].

1) MEDIAN FREQUENCY BALANCING

At present, the common method used in the image semantic segmentation task is to utilize the cross-entropy loss function to train the model. The cross-entropy loss function CE_{loss} is as follows:

$$CE_{loss} = -\frac{1}{N} \sum_{n=1}^{N} \sum_{c=1}^{C} l_c^{(n)} \cdot \log(p_c^{(n)}), \tag{9}$$

where N is the number of samples in a mini-batch, $p_c^{(n)}$ is the softmax probability of sample n being in class c, $l_c^{(n)}$ corresponds to the label of sample n for class c when the label is given in one-hot encoding, and C is the set of all of the classes.

However, as this loss is computed by the summation over all of the pixels, it does not account well for the imbalanced classes [6]. In order to consider the imbalanced classes, a common method is to introduce a weighting factor. Inspired by Eigen and Fergus [24], the loss of the classes is weighted using the median frequency balancing. Median frequency balancing weights the class loss by the ratio of the median class frequency in the training set and the actual class frequency. The cross-entropy loss function weighted by the median frequency balancing MFB_CE_{loss} is as follows:

$$MFB\_CE_{loss} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{c=1}^{C} w_c \cdot l_c^{(n)} \cdot \log(p_c^{(n)}), \quad (10)$$

where

$$w_c = \frac{median(f_c \mid c \in C)}{f_c}. \quad (11)$$

w_c is the class weight for class c, f_c is the frequency of pixels in class c, and $median(f_c)$ is the median value of each f_c.

2) MFB_$Focal_{loss}$ LOSS FUNCTION

Although the introduction of a weighting factor balances the importance of positive and negative samples, it does not distinguish between the easy and hard samples. Inspired by Lin *et al.* [23], the focal loss function was introduced to reduce the burden of the easy samples, thus putting more focus on the hard samples. The focal loss introduces a factor $(1\text{-}p_c^{(n)})^2$ based on the standard cross-entropy loss, and the focal loss function weighted by the median frequency balancing MFB_Focal_{loss} is as follows:

$$MFB\_Focal_{loss} = (1 - p_c^{(n)})^2 \cdot MFB\_CE_{loss} \quad (12)$$

The factor $(1 - p_c^{(n)})^2$ reduces the loss contribution from the easy samples and extends the range in which an example receives a low loss. For instance, when $p_c^{(n)} = 0.9$, the factor $(1\text{-}p_c^{(n)})^2 = 0.01$, MFB_Focal_{loss} would have a 100× lower loss compared with that of MFB_CE_{loss}. However, when $p_c^{(n)} = 0.1$, the factor $(1\text{-}p_c^{(n)})^2 = 0.81$, so MFB_Focal_{loss} would put more focus on the hard samples.

IV. EXPERIMENTS AND ANALYSIS

The International Society for Photogrammetry and Remote Sensing (ISPRS) meets every 4 years. The dataset used to evaluate our proposed method was the 23rd ISPRS Vaihingen 2D semantic labeling contest dataset [25]. The DenseU-Net proposed in this paper was compared with the U-Net [13] and Hourglass-Shaped Network (HSN) [19]. It is worth noting that the original U-Net was designed and tested on biomedical images. We strictly followed the U-Net design and used the Vaihingen dataset to train our model from scratch. The HSN uses the cross-entropy loss function weighted by the median frequency balancing MFB_CE_{loss} for experimentation. By using the overlap inference (OI) to systematically improve the accuracy of each class, post-processing using weighted belief propagation (WBP) further improved the overall accuracy. According to the rules of the contest, the F1-score of each class, average F1-score, and overall accuracy were selected for evaluation.

A. DATASET

We used the Vaihingen dataset [25] for our experiments. The dataset consists of 33 high resolution true ortho photo (TOP) images of varying size, ranging from approximately 3 million to 10 million pixels each. The image patch was taken of Vaihingen with a ground sampling distance of 9 cm. The ground truth images (GT images) were available for 16 of the 33 images, in which all of the pixels were labeled by 1 of 6 classes, namely Impervious Surfaces, Building, Low Vegetation, Tree, Car, or Clutter/Background. The "Clutter/Background" class includes water bodies and other objects (e.g., containers, tennis courts, and swimming pools), and it only accounted for a very small number of pixels, as shown in Table 2 and Figure 4. Therefore, just as with the HSN [19], we ignored the "Clutter/Background" class when evaluating the results.

TABLE 2. The pixel number of each class in the training set.

Class	Impervious surfaces	Building	Low vegetation	Tree	Car	Clutter/ Background
Pixels Number	15,932,837	14,647,182	11,008,085	12,118,796	666,618	510,494

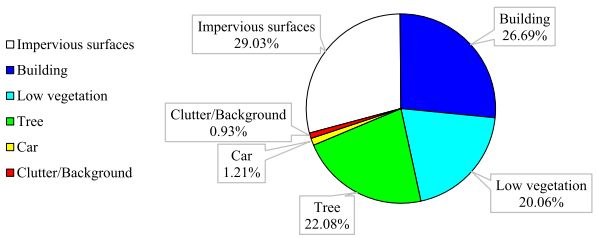


FIGURE 4. The proportion of pixels for each class in the training set.

We found that the Vaihingen dataset provided 16 ground truth images that ignored the boundary pixels (erGT images) corresponding to the GT images. To reduce the effect of the class boundaries, the class boundaries were eroded with a disk of radius 3 and ignored during evaluation, as specified by the ISPRS.

For fairness, and to follow the example of HSN [19], 16 GT images were divided into a training and test set. The training set consisted of 11 images (area: 1, 3, 5, 7, 13, 17, 21, 23, 26, 32, and 37) and the test set consisted of 5 images (area: 11, 15, 28, 30, and 34).

In this paper, 11 TOP images from the training set and corresponding GT images were cut into 256 × 256 images,

TABLE 3. Performance of the different models.

	Methods	Impervious surfaces	Building	Low vegetation	Tree	Car	Ave. F1	Overall Acc.
GT	HSN [19]	87.57	92.20	75.03	84.44	75.16	82.88	84.92
	HSN+OI [19]	88.01	92.37	75.83	84.86	76.50	83.51	85.38
	HSN+OI+WBP [19]	88.00	92.34	75.92	84.86	75.95	83.41	85.39
	U-Net+CE_{loss}	85.82	90.51	73.62	83.33	67.24	80.10	83.46
	U-Net+MFB_Focal_{loss}	85.64	90.31	72.86	83.25	76.52	81.72	83.21
	DenseU-Net+CE_{loss}	87.77	92.42	75.89	84.36	83.21	84.73	85.28
	DenseU-Net+MFB_Focal_{loss}	88.18	92.50	76.23	84.63	83.23	84.95	85.63
erGT	HSN [19]	90.89	94.51	78.83	87.84	81.87	86.79	88.32
	HSN+OI [19]	91.32	94.66	79.73	88.30	83.60	87.52	88.79
	HSN+OI+WBP [19]	91.34	94.67	79.83	88.31	83.59	87.55	88.82
	U-Net+CE_{loss}	88.92	92.62	77.45	86.70	75.54	84.24	86.75
	U-Net+MFB_Focal_{loss}	88.84	92.40	76.70	86.56	82.68	85.44	86.50
	DenseU-Net+CE_{loss}	90.89	94.57	79.77	87.74	90.83	88.76	88.57
	DenseU-Net+MFB_Focal_{loss}	91.30	94.64	80.17	87.99	90.96	89.01	88.92

and 50% of the neighboring images overlapped. Each cut image and its corresponding GT image were rotated (0°, 90°, 180°, and 270°), and then each rotated image was horizontally flipped, yielding 8 augmentations per cut image.

B. EVALUATION INDEX

To evaluate our method, we used the F1-score as an evaluation criterion for measuring the accuracy of each class and used the percentage of pixels that predict the correct class as the overall accuracy. Both values were between 0 and 1. The closer the value was to 1, the higher its accuracy. The 2 parameters used to calculate the F1-score involve *precision* and *recall*, which are respectively defined as

$$precision(c) = \frac{TP}{P} \times 100\%, \tag{13}$$

$$recall(c) = \frac{TP}{C} \times 100\%, \tag{14}$$

where TP indicates that the model correctly predicts the pixel number of class c, P represents the total pixel number in which the model predicts the samples as class c, and C is the total pixels number of class c in the sample.

The *precision* is the percentage of the correct results in the total results predicted by the model. The *recall* is the percentage of the correct results predicted by the model in the truth label of the sample. The F1-score also takes into account the *precision* and *recall* of the model, and its definition is as follows:

$$F1 = \frac{2 \times precision(c) \times recall(c)}{precision(c) + recall(c)} \times 100\%. \tag{15}$$

The overall accuracy is the percentage of pixels that predict the correct class, defined as

$$Accuracy = \frac{T}{A} \times 100\%, \tag{16}$$

where T represents the pixels number of the predict correct class, and A is the total number of all of the pixels.

C. RESULTS

In the Vaihingen dataset, the ''car'' class is difficult to handle because the ''car'' class is a relatively small object compared to other classes. As shown in Table 2 and Figure 4, the pixel number of the ''car'' class is much smaller than that of other classes, and there are large differences between the classes. The diversity of the car color in the images also leads to large differences within the class. Table 3 shows the experimental results of different models in the Vaihingen dataset. As shown in Table 3, DenseU-Net+MFB_Focal_{loss} is superior to other models in terms of its F1-score for each class, average F1-score, and overall accuracy. In order to verify the validity of MFB_Focal_{loss}, we did a comparative experiment using U-Net+CE_{loss} and U-Net+MFB_Focal_{loss}. In the case where the boundary pixels were taken into account, U-Net+MFB_Focal_{loss} shows an increase of 9.28% in the ''car'' class compared to U-Net+CE_{loss}, while still maintaining a good overall accuracy. In order to verify the performance of DenseU-Net, we retrained DenseU-Net+CE_{loss} and DenseU-Net+MFB_Focal_{loss}. As shown in Table 3, the F1-score of the ''car'' class for DenseU-Net+CE_{loss} increased by 15.97% compared with that of U-Net+CE_{loss}. The F1-score of other classes improved to varying degrees, while the average F1-score and overall accuracy increased by 4.63% and 1.82%, respectively. The F1-score of the ''car'' class for DenseU-Net+MFB_Focal_{loss} increased by 6.71% compared with that of U-Net+MFB_Focal_{loss}, while the average F1-score and overall accuracy increased by 3.23% and 2.42%, respectively.

Compared with HSN+OI+WBP, DenseU-Net+MFB_Focal_{loss} does not use any post-processing and achieves an overall accuracy of 85.63%, which is still better than the 85.39% of HSN+OI+WBP. The F1-score of each class improved to varying degrees, especially the ''car'' class. The F1-score of the ''car'' class for DenseU-Net+MFB_Focal_{loss} exceeds 7.28%.

In the case where the border pixels were ignored (erGT), all models performed better than in the case in which the border pixels were taken into account (GT). This result is due to the ambiguities around the object boundaries. As shown in Table 3, in the case where the border pixels were ignored (erGT), DenseU-Net+MFB_Focal_{loss} achieves an overall accuracy of 88.92%, which is still better than HSN+OI+WBP.

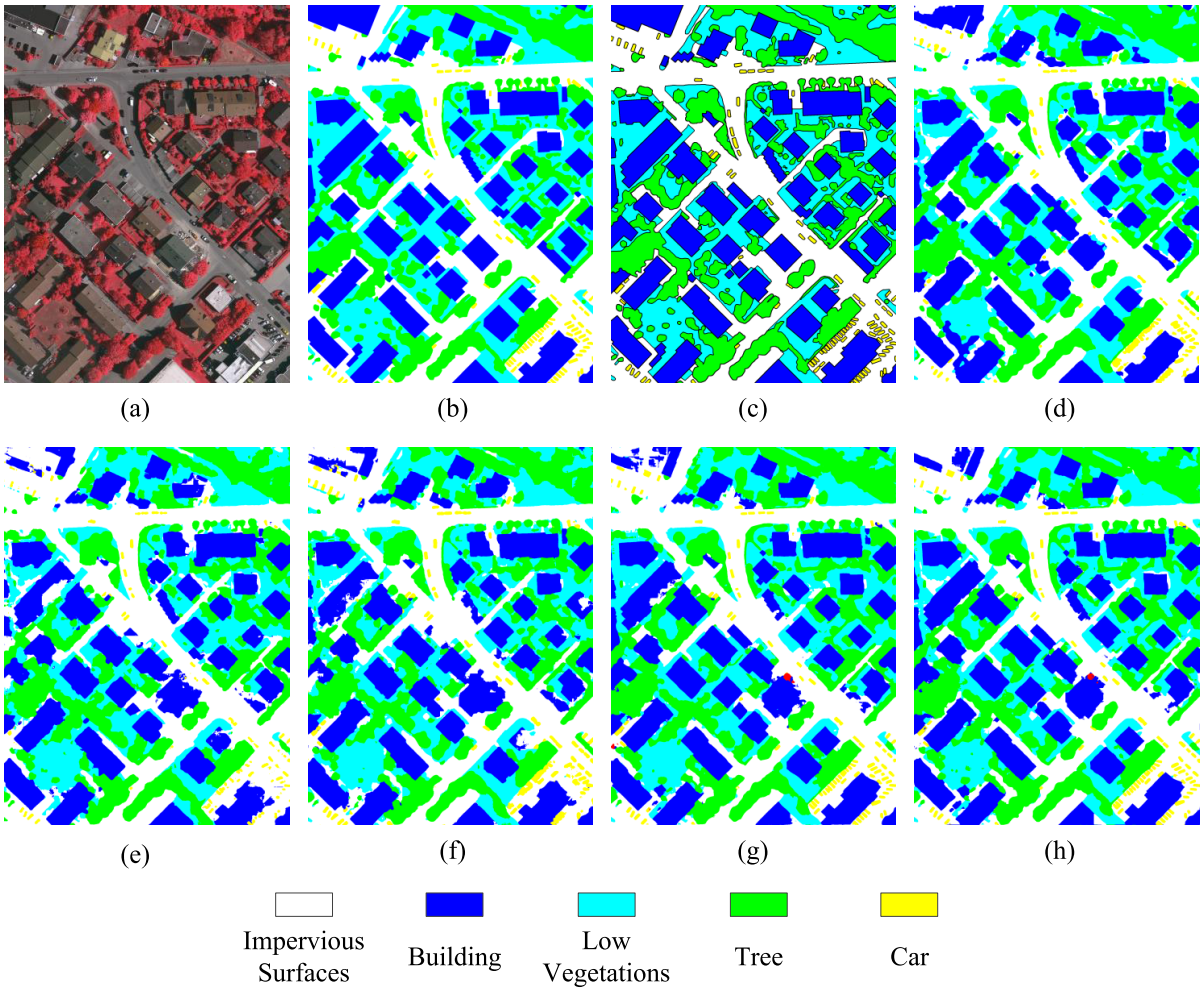


FIGURE 5. Visual comparison of the overall semantic segmentation results in the Vaihingen dataset. (a) TOP. (b) GT. (c) erGT. (d) HSN+OI+WBP. (e) U-Net+CE_{loss}. (f) U-Net+MFB_Focal_{loss}. (g) DenseU-Net+CE_{loss}. (h) DenseU-Net+MFB_Focal_{loss}.

DenseU-Net architecture extracts CNN features using continuous DownBlocks during the contracting path, which enables the architecture to obtain context information, and restores the resolution of image layer by layer using continuous UpBlocks during the expanding path, which enables the architecture to obtain position information. Meanwhile, DenseU-Net applies its symmetrical structure to fuse the detail features in shallow layers and the abstract semantic features in deep layers, which contributes to feature extraction of small object class. MFB_Focal_{loss} balances the importance between positive and negative samples by means of median frequency balancing, and a factor is introduced to reduce the relative loss for well-classified examples, putting more focus on hard-misclassified examples, optimizing the training process and helping feature extraction of hard-misclassified examples.

Figure 5 shows a visual comparison of the overall semantic segmentation results in the Vaihingen dataset for different models. The overall accuracy for DenseU-Net+MFB_Focal_{loss} is better than that of the other models. Figure 6 shows a visual comparison of the local semantic segmentation results in the Vaihingen dataset for different models. DenseU-Net+MFB_Focal_{loss} performs better than other models in the ''car'' class. The segmentation effect is more accurate and the boundaries are smoother than what is found in other models. For instance, for the ''Low vegetation'' class in the upper right corner of image, HSN+OI+WBP and U-Net (U-Net+CE_{loss} and U-Net+MFB_Focal_{loss}) are misclassified as ''Tree'', but the results of DenseU-Net (DenseU-Net+CE_{loss} and DenseU-Net+MFB_Focal_{loss}) have been greatly improved. For the ''Building'' class in the center of image, the segmentation results of DenseU-Net (DenseU-Net+CE_{loss} and DenseU-Net+MFB_Focal_{loss}) are significantly better than that of U-Net (U-Net+CE_{loss} and U-Net+MFB_Focal_{loss}). It is proved that DenseU-Net can effectively improve the segmentation performance. For the ''Low vegetation'' class in the upper left and lower left corners of image, DenseU-Net+CE_{loss} and U-Net+CE_{loss} are misclassified as ''Tree'', but DenseU-Net+MFB_Focal_{loss} and U-Net+MFB_Focal_{loss} get more accurate segmentation

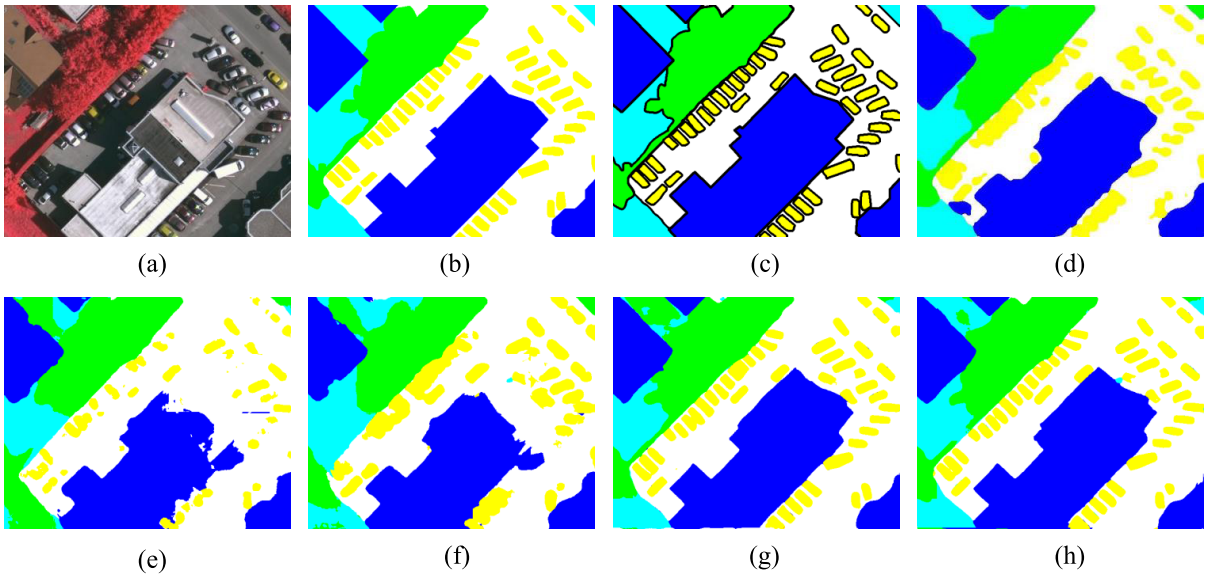


FIGURE 6. Visual comparison of the local semantic segmentation results in the Vaihingen dataset. (a) TOP. (b) GT. (c) erGT. (d) HSN+OI+WBP. (e) U-Net+CE_{loss}. (f) U-Net+MFB_Focal_{loss}. (g) DenseU-Net+CE_{loss}. (h) DenseU-Net+MFB_Focal_{loss}.

results. It is proved that MFB_Focal_{loss} can effectively improve the segmentation performance. For the small object class ''Car'', it can be clearly seen that the segmentation performance of DenseU-Net (DenseU-Net+CE_{loss} and DenseU-Net+MFB_Focal_{loss}) are better than U-Net (U-Net+CE_{loss} and U-Net+MFB_Focal_{loss}). Meanwhile, compared with HSN+OI+WBP, DenseU-Net+MFB_Focal_{loss} gets more accurate segmentation results and smoother boundary.

V. CONCLUSIONS

To address the class imbalance problem of semantic segmentation in urban remote sensing images, an end-to-end deep convolutional neural network (DenseU-Net) architecture for pixel-wise urban remote sensing image segmentation is proposed in this paper. The DenseU-Net applies the continuous DownBlocks in the contracting path to extract the CNN features, and applies the continuous UpBlocks in the expanding path to restore the resolution of the image. This method preserves the details such as the color and texture of the images. DenseU-Net applies its symmetrical structure to fuse the detail features in shallow layers and the abstract semantic features in deep layers. A focal loss function weighted by the median frequency balancing (MFB_Focal_{loss}) is proposed that facilitates the feature extraction of small object classes. The experiment on the Vaihingen dataset shows that DenseU-Net+MFB_Focal_{loss} can identify the small objects class ''car'' well, while still maintaining good overall accuracy, which is superior to the other models tested both numerically and visually.

REFERENCES

[1] H. Ghassemian, ''A review of remote sensing image fusion methods,'' *Inf. Fusion*, vol. 32, pp. 75–89, Nov. 2016.

[2] O. Regniers, L. Bombrun, V. Lafon, and C. Germain, ''Supervised classification of very high resolution optical images using wavelet-based textural features,'' *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 6, pp. 3722–3735, Jun. 2016.

[3] D. Tuia, N. Courty, and R. Flamary, ''Multiclass feature learning for hyperspectral image classification: Sparse and hierarchical solutions,'' *ISPRS J. Photogramm. Remote Sens.*, vol. 105, pp. 272–285, Jul. 2015.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, ''Imagenet classification with deep convolutional neural networks,'' in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.

[5] R. S. Dong, D. Q. Cheng, and F. Y. Li, ''Aggregating deep convolutional features for image retrieval using multi-regional cross weighting,'' (in Chinese), *J. Comput.-Aided Des. Comput. Graph.*, vol. 30, no. 4, pp. 658–665, Apr. 2018.

[6] M. Kampffmeyer, A. B. Salberg, and R. Jenssen, ''Semantic segmentation of small objects and modeling of uncertainty in urban remote sensing images using deep convolutional neural networks,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2016, pp. 1–9.

[7] M. Kampffmeyer, A.-B. Salberg, and R. Jenssen, ''Urban land cover classification with missing data using deep convolutional neural networks,'' in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2017, pp. 5161–5164.

[8] A. Estabrooks, T. H. Jo, and N. Japkowicz, ''A multiple resampling method for learning from imbalanced data sets,'' *Comput. Intell.*, vol. 20, no. 1, pp. 18–36, 2004.

[9] M. D. Zeiler and R. Fergus, ''Visualizing and understanding convolutional networks,'' in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 818–833.

[10] K. Simonyan and A. Zisserman, ''Very deep convolutional networks for large-scale image recognition,'' 2014, *arXiv:1409.1556*. [Online]. Available: https://arxiv.org/abs/1409.1556

[11] C. Szegedy *et al.*, ''Going deeper with convolutions,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1–9.

[12] J. Long, E. Shelhamer, and T. Darrell, ''Fully convolutional networks for semantic segmentation,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 3431–3440.

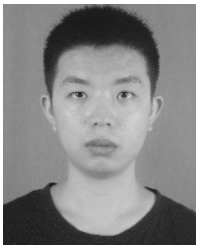
[13] O. Ronneberger, P. Fischer, and T. Brox, ''U-Net: Convolutional networks for biomedical image segmentation,'' in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent*, 2015, pp. 234–241.

[14] V. Mnih and G. E. Hinton, ''Learning to label aerial images from noisy data,'' in *Proc. 29th Int. Conf. Mach. Learn. (ICML)*, 2012, pp. 567–574.

[15] V. Mnih, ''Machine learning for aerial image labeling,'' Ph.D. dissertation, Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, 2013.

[16] S. Saito, T. Yamashita, and Y. Aoki, "Multiple object extraction from aerial imagery with convolutional neural networks," *Electron. Imag.*, vol. 10, no. 9, pp. 1–9, 2016.
[17] M. Volpi and D. Tuia, "Dense semantic labeling of subdecimeter resolution images with convolutional neural networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 2, pp. 881–893, Feb. 2016.
[18] V. Iglovikov, S. Mushinskiy, and V. Osin, "Satellite imagery feature detection using deep convolutional neural network: A kaggle competition," 2017, *arXiv:1706.06169*. [Online]. Available: https://arxiv.org/abs/1706.06169
[19] Y. Liu, D. M. Nguyen, N. Deligiannis, W. Ding, and A. Munteanu, "Hourglass-shapenetwork based semantic segmentation for high resolution aerial imagery," *Remote Sens.*, vol. 9, no. 6, p. 522, 2017.
[20] X. Gao *et al.*, "An end-to-end neural network for road extraction from remote sensing imagery by multiple feature pyramid network," *IEEE Access*, vol. 6, pp. 39401–39414, 2018.
[21] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.
[22] N. V. Chawla, N. Japkowicz, and A. Kotcz, "Special issue on learning from imbalanced data sets," *ACM Sigkdd Explor. Newslett.*, vol. 6, no. 1, pp. 1–6, Jun. 2004.
[23] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published.
[24] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 2650–2658.
[25] *ISPRS Vaihingen 2D Semantic Labeling Dataset*. Accessed: Apr. 5, 2018. [Online]. Available: http://www2.isprs.org/commissions/comm3/wg4/2d-sem-label-vaihingen.html

RONGSHENG DONG received the B.S. degree from the China University of Geosciences, Wuhan, China, in 1989. He is currently a Professor with the School of Computer Science and Information Security, Guilin University of Electronic Technology. His current research interests include graph data, social computing, and protocol engineering. He has received research grants from NSFC and GXNSF. He is currently a member of the Advisory Commission on Teaching Computer Curriculum in Colleges and Universities under the Ministry of Education and the Education Commission of China Computer Federation.



XIAOQUAN PAN is currently pursuing the master's degree with the School of Computer Science and Information Security, Guilin University of Electronic Technology. His research interests include deep learning, computer vision, and semantic segmentation.

FENGYING LI received the B.S. and M.S. degrees from the Guilin University of Electronic Technology, China, in 1994 and 2002, respectively, and the Ph.D. degree from Xidian University, in 2011. She is currently an Associate Professor with the School of Computer Science and Information Security, Guilin University of Electronic Technology. Her research interests include symbolic computing, formal method, and Petri net theory.