ISPRS Journal of Photogrammetry and Remote Sensing 156 (2019) 1–13

Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

journal homepage: www.elsevier.com/locate/isprsjprs

TreeUNet: Adaptive Tree convolutional neural networks for subdecimeter aerial image segmentation

Kai Yue, Lei Yang, Ruirui Li*, Wei Hu, Fan Zhang, Wei Li

Beijing University of Chemical Technology, Beijing, China
College of Information Science & Technology, North Third Ring Road 15, Chaoyang District, Beijing 100029, China

ARTICLE INFO

Keywords:
Aerial imagery
Semantic segmentation
Tree structures
Adaptive network
ISPRS
CNN

ABSTRACT

Fine-grained semantic segmentation results are typically difficult to obtain for subdecimeter aerial imagery segmentation as a result of complex remote sensing content and optical conditions. Recently, convolutional neural networks (CNNs) have shown outstanding performance on this task. Although many deep neural network structures and techniques have been applied to improve accuracy, few have attended to improving the differentiation of easily confused classes. In this paper, we propose TreeUNet, a tool that uses an adaptive network to increase the classification rate at the pixel level. Specifically, based on a deep semantic model infrastructure, a Tree-CNN block in which each node represents a ResNeXt unit is constructed adaptively in accordance with the confusion matrix and the proposed TreeCutting algorithm. By transmitting feature maps through concatenating connections, the Tree-CNN block fuses multiscale features and learns best weights for the model. In experiments on the ISPRS two-dimensional Vaihingen and Potsdam semantic labelling datasets, the results obtained by TreeUNet are competitive among published state-of-the-art methods. Detailed comparison and analysis show that the improvement brought by the adaptive Tree-CNN block is significant.

1. Introduction

Semantic segmentation consists in the assignment of a semantic label (land-cover or land-use class) to every pixel of an image. Recently, highly developed remote sensing techniques have been able to provide very-high-resolution (VHR) aerial images with a ground sampling distance of 5–10 cm in the spatial or spectral domain. As a result, small objects such as cars and buildings are distinguishable and can be segmented. When processing ultra-high resolution data, most previous methods have relied on supervised classifiers that were trained on hand-crafted feature sets that describe the image content locally. The extracted high-dimensional representation is assumed to contain sufficient information to classify pixels. In fact, these features depend on a specific feature extraction method whose parameters and performance on the data in question were previously unknown.

Deep CNNs (DCNNs) have been extremely successful in many high-level computer vision tasks, ranging from image classification (Krizhevsky et al., 2012) to object detection, visual recognition and semantic segmentation. DCNNs address trainable tasks in an end-to-end fashion, which usually involves joint learning of a series of feature extractions from raw input data to a final, task-specific output. DCNNs have also been applied to remote sensing. A fully convolutional network (FCN) (Long et al., 2015) is a classic DCNN specifically designed for semantic segmentation. Sherrah et al. (Wilkinson et al., 2011) and Liu et al. (2017) applied FCNs to remote sensing imagery and published their results on the ISPRS two-dimensional (2D) benchmark. Hourglass networks with skip connections, including DeconvNet (Noh et al., 2015), SegNet (Badrinarayanan et al., 2017) and U-Net (Ronneberger et al., 2015) were then applied to aerial imagery labelling (Audebert et al., 2018; Marmanis et al., 2018; Volpi and Tuia, 2017; Wang et al., 2017), obtaining higher accuracy.

As is often observed, aerial imagery is characterised by complex data properties in the form of heterogeneous and easily mixed classes. Even for the existing well-functioning DCNN structures, semantic labelling on aerial imagery is still difficult and requires better comprehension of the image context. Some studies have investigated context using multitask learning (Marmanis et al., 2018; Volpi and Tuia, 2018), multiscale feature aggregation (Maggiori et al., 2017) or multi-model fusion (Audebert et al., 2018; Liu et al., 2017). Differently but not in contradiction with the above methods, this paper proposes an adaptive

Abbreviations: CNN, convolutional neural networks; CRF, conditional random fields; DSM, digital surface model; FCN, fully convolutional network; NDVI, normalised digital vegetation index; OA, overall accuracy; RF, random forest; VHR, very-high-resolution

* Corresponding author at: Beijing University of Chemical Technology, Beijing, China.
E-mail address: liruirui@mail.buct.edu.cn (R. Li).

https://doi.org/10.1016/j.isprsjprs.2019.07.007
Received 29 April 2018; Received in revised form 12 July 2019; Accepted 15 July 2019
Available online 01 August 2019

network called TreeUNet that contains an automatically constructed Tree-CNN block. The Tree-CNN block together with new skip connections is designed for multiclass labelling of easily confused categories. TreeUNet extends basic segmentation networks such as DeepUNet (Li et al., 2018) or DeepLab v3+ to achieve an impressive improvement in accuracy. In summary, we offer the following contributions:

1) An end-to-end deep neural network architecture that connects a segmentation module and a Tree-CNN block is proposed, resulting in better results on the ISPRS Vaihingen and Potsdam datasets. On the basis of DeepUNet, TreeUNet obtained competitive results among state-of-the-art methods.
2) This paper introduces an automatic method for constructing the adaptive Tree-CNN block. The learned Tree-CNN network structure can assist in differentiating easily confused classes. As far as we know, it is the first adaptive CNN network to be applied to semantic labelling tasks of remote sensing images.
3) A group of experiments are performed on the ISPRS Vaihingen and Potsdam datasets, making qualitative and quantitative comparisons among state-of-the-art methods and deep models. The results of DeepUNet, DeepLab v3+ and TreeUNet are submitted to the ISPRS community.

The remainder of this paper is organised as follows. In Section 2, we introduce research topics that are related to remote sensing semantic labelling, deep learning and adaptive networks. The method proposed in this paper is described in Section 3. Section 4 shows training details that cover data preprocessing, training parameters and image stitching. Section 5 discusses the experimental situation and analyses the results. The last section draws conclusions.

2. Related work

2.1. Semantic segmentation for remote sensing imagery

Semantic segmentation on remote sensing images involves the classification of houses, vehicles, roads, vegetation, oceanic ice and more at pixel-level precision. The remote sensing images can come from aerial imagery or spectroscopy sensors. Early studies were primarily based on graph theory (Boykov, 2001; Di et al., 2017; Grady, 2006; Grady and Schwartz, 2004; Shi and Malik, 2000) using unsupervised learning. In 2009, Ye and Wang (2009) proposed a new segmentation method for remote sensing imagery that combines a minimum spanning tree algorithm with Mumford Shah theory. Cui and Zhang (2011) proposed a multiscale and multilevel segmentation method based on a minimum spanning tree in 2011 that performed well in VHR remote sensing image segmentation. For supervised learning, most segmentation methods have learned with features selected by hand (Ouma et al., 2008; Reis and Taşdemir, 2011; Wang et al., 2016; Yu et al., 2016). These features are typically complex and can express only low- or mid-level descriptions.

The development of remote sensing technologies has made it easy to obtain a large number of VHR remote sensing images. These images contain a wealth of contextual information, making most traditional segmentation methods unsuitable. CNNs initially learn semantic representation of a pixel through patch-based training. The FCN introduced by Long et al. at the University of California, Berkeley, is a breakthrough. It replaces all fully connected layers with convolutional neurons to allow arbitrary image size segmentation. Dilated convolution (Noh et al., 2015) provides a larger receptive field under the same computational conditions and could be used as a pooling operation to reduce dimensionality. Substantial work has resulted in proposals to improve dilated convolution, such as Atrous Spatial Pyramid Pooling (Chen et al., 2017) and fully connected conditional random fields (CRFs) (Chen et al., 2018a). Inspired by encoder–decoder architectures, modified hourglass networks (Audebert et al., 2018; Marmanis et al., 2018; Volpi and Tuia, 2017; Wang et al., 2017) were proposed to handle pixel-level segmentation on VHR remote sensing images. Some proposed methods learn the features in a multitask manner to improve understanding of the context. Others take advantage of multiscale technology to aggregate features of different resolutions. Remote sensing images are typically stored in a heterogeneous manner. FuseNet (Audebert et al., 2018) improves accuracy by fusing features from multimodal data.

2.2. Adaptive networks in multiclass labelling

In early research, adaptive neural networks helped to classify images through relaxed hierarchy structures in which a subset of confusing classes could be ignored (Deng et al., 2011; Gao and Koller, 2011; Griffin and Perona, 2008). The principal difference between various existing methods is in the manner that the hierarchy is built. One common fundamental problem for these methods is that the assumption regarding the easy separability of a binary partition of classes at each node is not valid when there are many classes (Marszałek and Schmid, 2008). Thus, these methods do not scale well in regard to classification accuracy. Since deep learning has become a popular research topic, many CNN-based adaptive network methods have been proposed. Srivastava and Salakhutdinov (2013) were the first to introduce a category hierarchy in CNN-based methods. In these tree-like hierarchical CNN models, the upper nodes use extracted common features to classify images into superclasses, whereas the deeper nodes address finer features and perform further discrimination. Hierarchical DCNNs (Yan et al., 2015) use common features shared between images to build a hierarchical CNN model for visual recognition.

The tree structures discussed above are somewhat similar to our proposed tree structure. These networks differ from ours in being built before training, with the structure staying invariable in the process. Some dynamic methods have also been explored. Adaptive neural trees (ANTs) (Tanno et al., 2018) are used to grow an architecture adaptively from primitive modules. Xiao et al. (2014) introduced a method that grows a tree-shaped network to accommodate new classes. Lifelong learning (Yoon et al., 2017) requires a model to adapt to new tasks while retaining its performance on previous tasks.

Some methods that can inherit previous training knowledge have been explored. ‘Learning without Forgetting’ (Li and Hoiem, 2016) is one such method; it uses only new task data to train the network while preserving the original capabilities. Progressive neural networks (Rusu et al., 2016) learn to solve complex sequences of tasks by leveraging prior knowledge with lateral connections. Researchers have also designed spatially adaptive networks (Bengio et al., 2015; Figurnov et al., 2017) in which nodes in a layer are activated selectively. Others have developed cascade approaches (Leroux et al., 2017; Odena et al., 2017) that allow early exits based on confidence feedback.

Our proposed method constructs a Tree-CNN block based on knowledge learned from progressive training. It does not predefine any structures and can be constructed automatically. The confusion matrix used to construct the tree is considered to be the recording memory. To the best of our knowledge, our work is the first to use both adaptive hierarchies and a deep neural network in a unified deep learning structure for remote sensing semantic labelling. ANTs (Tanno et al., 2018) involving a similar unified deep learning structure that combines decision trees and CNNs have also been proposed. That work became available on arXiv 5 months later than ours and does not perform semantic segmentation on remote sensing images.

3. Proposed method

Learning semantic representations for pixels for multiclass labelling tasks on remote sensing images is difficult, because easily confused classes are usually adjacent or interlaced in distribution. On the basis of the observation that classes can share some features while also

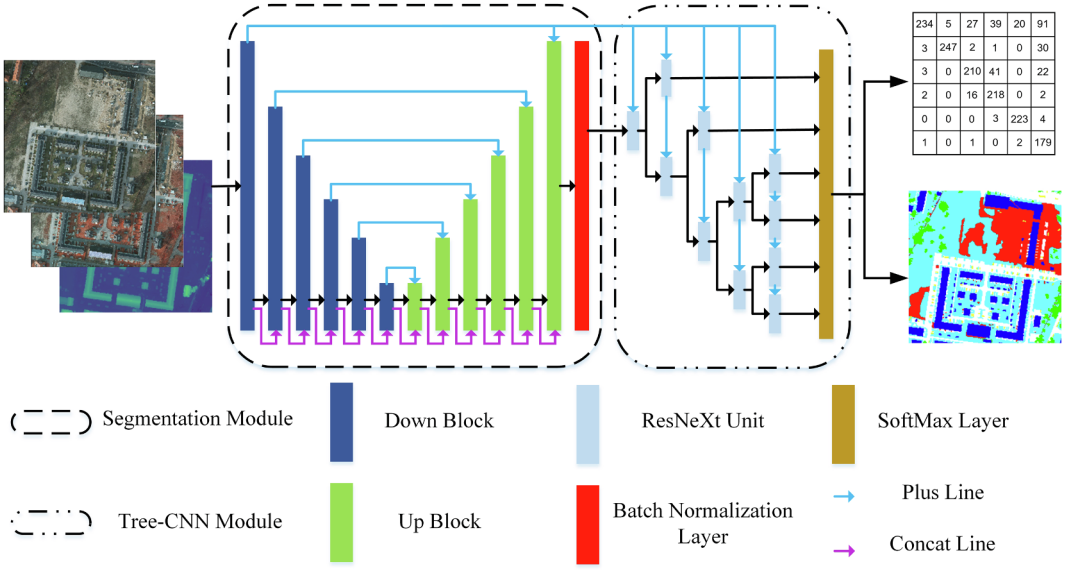

Fig. 1. The overview of TreeUNet framework.

possessing their own properties, we attempt to address the multiclass labelling problem by hierarchically organising the neural modules into a collection of subgroups and providing the easily confused classes more convolutional computations. To conduct this approach in an end-to-end fashion, we propose a novel CNN architecture called TreeUNet for semantic segmentation tasks on remote sensing images. The overall framework is illustrated in Fig. 1.

The TreeUNet architecture consists primarily of three parts: the segmentation module, the Tree-CNN block, and the concatenating connections. We use hourglass encoder–decoder networks for the segmentation module. Then, the Tree-CNN block is built using an automatic construction method in accordance with the confusion degree among the classes. On the first pass, the method must train an initial model without the Tree-CNN block and compute the confusion matrix from the first-pass segmentation results. In subsequent passes, the Tree-CNN block is added and updated after each iteration. Specifically, after each iteration's training is completed, a new confusion matrix is calculated in accordance with the segmentation results. The iterations are repeated until the structure of the Tree-CNN block no longer changes.

3.1. Segmentation module

As shown in Fig. 1, we initially use DeepUNet as the infrastructure network for pixelwise semantic segmentation. DeepUNet is based on VGG16 (Simonyan and Zisserman, 2014). It has two processing paths: the contracting and expanding paths (Fig. 2). In the contracting path, the DownBlock is used as the basic feature extractor. It contains two convolutional layers and one pooling layer. Symmetrically, in the expanding path, the UpBlock is used as the upsampling block. Features are passed from the DownBlock to the UpBlock of the same level and then concatenated to perform convolution and upsampling. Both the DownBlocks and the UpBlocks use the residual operations and skip connections, which allow DeepUNet to be superior to other similar CNN architectures, such as SegNet and DeconvNet.

For VHR remote sensing images, we also try deeper neural networks to extract more contextual information. We use the latest DeepLab v3+ as the infrastructure for the segmentation module. The DeepLab v3+ is illustrated in Fig. 3. It consists of an entry flow, a number of repeated Xceptions as the middle flow, an exit flow, an ASPP block and several decoder layers.

3.2. Tree-CNN block

As shown in Fig. 4, the construction of the Tree-CNN block starts with calculating the confusion matrix in accordance with the first-time segmentation results. The confusion matrix, also known as an error matrix, is a specific table layout that allows visualisation of the performance of an algorithm. Each row of the matrix represents the instances of a predicted class, whereas each column represents the instances of an actual class (or vice versa). With the confusion matrix, the following four steps are used to construct the Tree-CNN block:

(1) Calculate the lower triangular matrix;
(2) Build the undirected graph;
(3) Iterate the TreeCutting operation; and
(4) Construct the tree structure.

In confusion matrix A, we use a_{ij} to represent the element in row i, column j and b_{ij} to represent the element in the lower triangular confusion matrix B. According to the following formula, the lower triangular matrix B can be obtained by adding the values of the symmetric positions of the matrix A. Here, b_{ij} represents the confusion degree between class i and class j.

$$b_{ij} = \begin{cases} a_{ij} + a_{ji}, & i > j \\ 0, & \text{otherwise} \end{cases}$$

The lower triangular matrix can be seen as an adjacency matrix that is associated with an undirected graph. In the graph, the nodes represent the indices of the classes, and the values of the connections represent the weights of the confusions. The numbers 1 through 6 represent imp_surf, building, low_veg, tree, car and clutter, respectively.

The undirected graph is then transformed into a binary tree structure using either the minimum-cut algorithm or TreeCutting algorithm (see Appendix A). It was found that trees constructed using TreeCutting and those constructed using the minimum-cut algorithm are the same even though the procedures for constructing them are different. In fact, TreeCutting can be seen as a simple approximation of the minimum-cut algorithm for undirected complete graphs.

The Tree-CNN block is the core of TreeUNet, which can enhance the results of the previous semantic segmentation. It is part of our TreeSegNet network. The input of the Tree-CNN block is the feature map extracted using the previous segmentation module, and its output is the final segmentation result through the SOFTMAX layer. The ISPRS

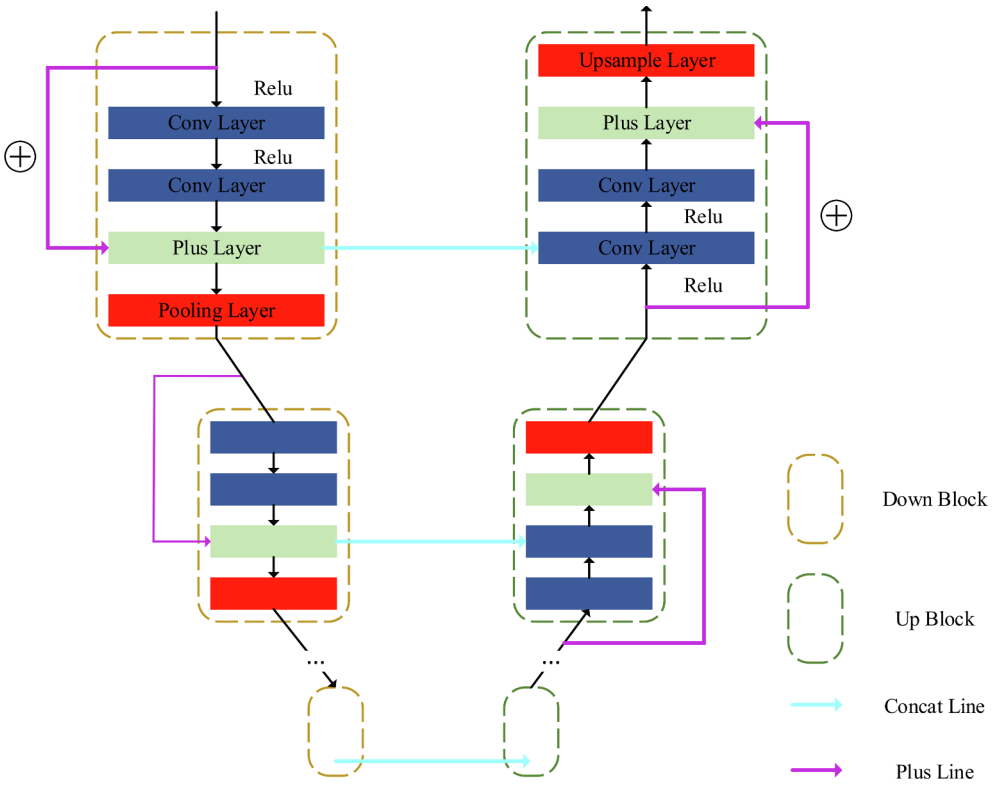

Fig. 2. DeepUNet network structure with the contracting and expanding paths.

datasets have six categories. Thus, the Tree-CNN block is illustrated with a binary tree that has six leaf nodes in Fig. 5. Each node is a ResNeXt unit shown on the right-hand side in Fig. 5. Through continuous optimisation of back propagation, features of the distinct classes tend to go through the shorter path with fewer convolutional layers. In other words, the most easily confusing classes tend to choose the path that contains more neural layers for further feature extraction. It is important to use the ResNeXt units for the Tree-CNN block. These units can both avoid the gradient vanishing problem caused by deeper neural layers and also save graphics memories by reducing the number of hyper-parameters. The features that enter the ResNeXt unit have two sources: the output of the previous ResNeXt unit and the concatenating connections (see Fig. 6).

Concatenating connections, a type of skip connections, are used to transmit features. The features output by the segmentation module are limited in size for the subsequent training of the Tree-CNN block. Thus, we connect the feature map output by the first convolutional layer to the input channels of the ResNeXt unit. This type of connection passes the same feature map to all ResNeXt units of the Tree-CNN block, reusing the information extracted previously. The number of features in the concatenating connections is relevant to the overall accuracy (OA). In the experiment, we tested 16, 32 and 64 features to determine the best degree of feature reuse.

TreeUNet is a full CNN. It has no fully connected layers. After the Tree-CNN block, all of the features are passed to a 1×1 convolutional layer, and the weights are updated by the SOFTMAX loss function before the output of predictions.

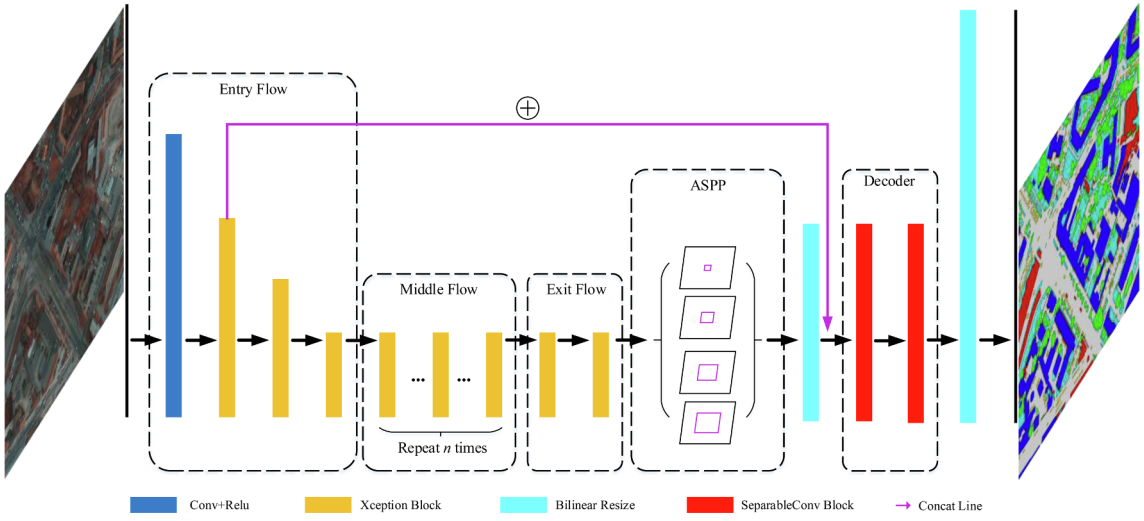

Fig. 3. DeepLab v3+: encoder–decoder with atrous convolution and ASPP.

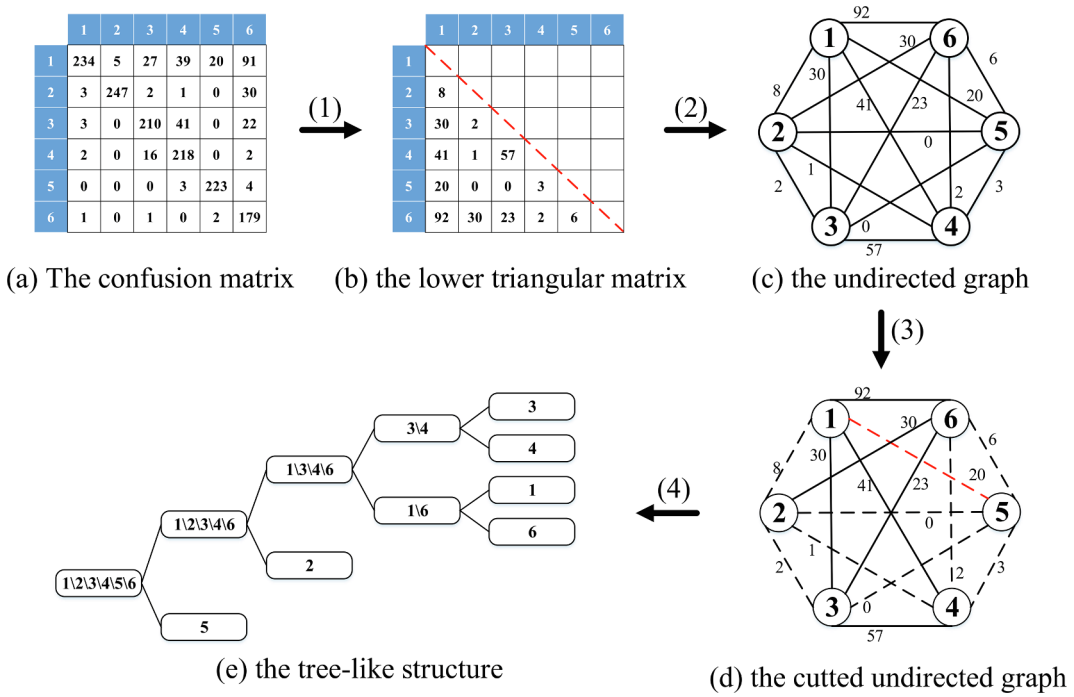

Fig. 4. Constructing a Tree-CNN block involves four steps: (1) calculate the lower triangular matrix b from the confusion matrix a; (2) build the undirected graph c; (3) iterate the TreeCutting operation; and (4) construct the tree structure e.

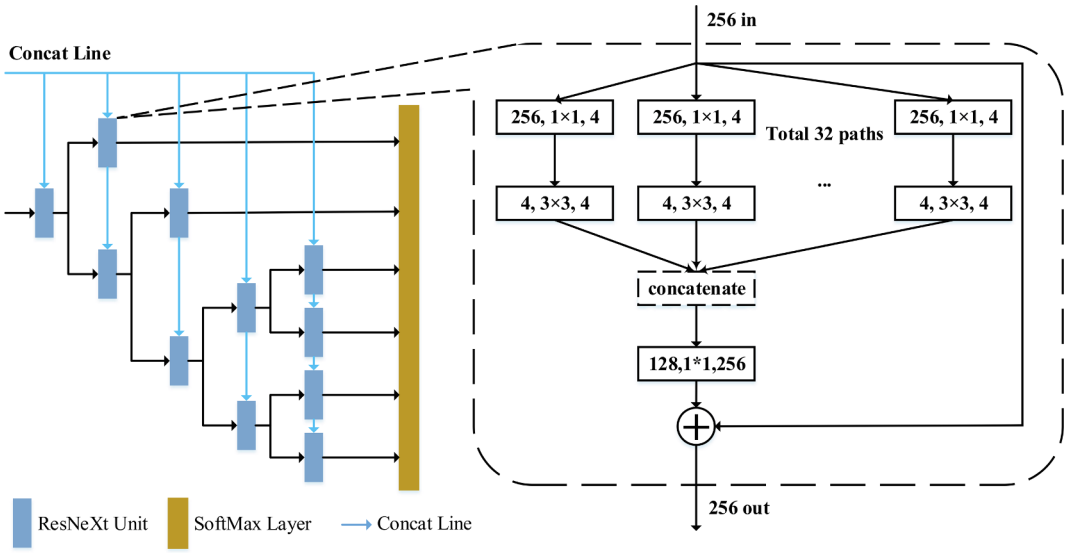

Fig. 5. Illustration of a Tree-CNN block. The blue lines are concatenating connections. The blue block is zoomed in on and its structure is shown on the right. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

4. Implementation details

4.1. Data preprocessing

The ISPRS Vaihingen and Potsdam 2D datasets contain multimodal data. Before entering TreeUNet, IRRG/RGB/digital surface model (DSM) images are assembled into multiple channels in a single image pattern.

Then, data enhancement is performed on the images using transformation, clipping and rotation. Since the aerial images are orthogonal images that were photographed from above, they are rotated by θ° repeatedly for 360° to maximise the number of training data images. In our experiments, we chose $\theta = 10$ to obtain a tradeoff between the training time and the training data. As a result, 36 images are obtained after the rotation for each VHR image. In according with the steps shown in Fig. 7, we calculate the largest horizontal aligned square in the rotated image and clip it.

4.2. Overlapping tiles

The obtained square images are still too large for training. We must clip the large images further into tiles and test the tiles one by one from the top left to the bottom right using a sliding window approach. For VHR images, we propose an overlapping tile strategy. Overlapping tiles are used not only because of GPU memory limitations but also because of higher segmentation result accuracy.

The pixels surrounded by yellow lines in Fig. 8 indicate the valid area to be predicted. The pixels between the red and the yellow lines

Fig. 6. Samples from 2D semantic labelling contest Potsdam dataset. From left to right, there are data for the (a) RGB, (b) IRRG, (c) DSM channels and (d) the corresponding ground truth.

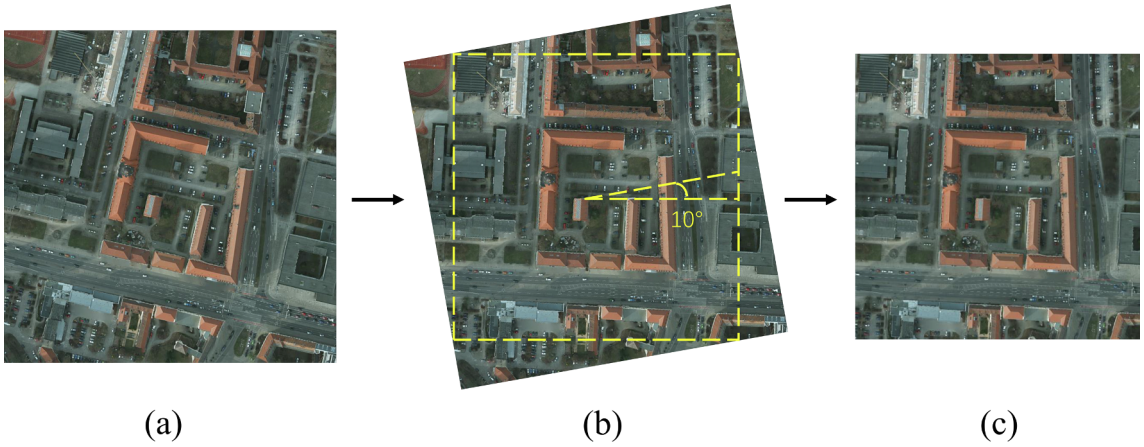

Fig. 7. The original image (a) is rotated around the centre by θ°, resulting in the rotated image (b), which is then clipped to get the largest horizontal aligned square (c).

Fig. 8. Schematic diagram of overlapping tiles. Prediction of the segmentation in the yellow area requires image data within the red area as input. Missing input data are extrapolated by mirroring as indicated by the white line. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

are called overlapping tiles or boundary fields; they provide contextual information to obtain a more accurate prediction. To predict the pixels in the border region of the image, the missing context is extrapolated by mirroring the input image. For the overlapping tiles, TreeUNet computes the weight using a 2D Gaussian function. The parameter σ is set to the default value 0.5. Pixels that are closer to the centre have greater weights for subsequent stitching. Through the use of a weighted sum, we composite the overlapping tiles and seamlessly stitch the entire segmented image.

$$g(x, y) = \frac{1}{2\pi\sigma^2} e^{-\left[\frac{(x-x_0)^2+(y-y_0)^2}{2\sigma^2}\right]}$$

4.3. Training

TreeUNet is implemented on MXNet, a GPU-friendly deep learning framework. The Gluon library provides tools for fast implementation of deep neural networks based on it.

To minimise the overhead and make maximum use of the GPU memory, we favour large input tiles over a large batch size. We chose 640×640 as the dimension of the input tiles. The epoch is initially set to 80. We use a high momentum (0.9). For the learning rate, the initial setting is 0.01. When the training is half completed, the learning rate is adjusted to 0.001. When the number of training steps reaches 3/4 of the total, the learning rate is adjusted again to 0.0001.

During the training process, the segmentation module and the Tree-CNN block are connected and trained as one unified network structure. In backpropagation, training errors are passed from the Tree-CNN block to the segmentation module. Note that the structure of the Tree-CNN block will be updated as the confusion matrix changes throughout the training process. However, at every iteration of the process, the structure of the Tree-CNN block remains invariable.

5. Experiments and analysis

In this section, the experimental environments, dataset descriptions, evaluation methods and experiments in both qualitative and quantitative comparisons are presented. Then, the proposed TreeUNet is analysed in detail using a serious of ablation experiments.

Table 1
Experimental environment.

CPU	Intel (R) Core (TM) i7-4790 K 4.00 Hz
GPU	2 * GeForce GTX1080 Ti
RAM	16 GB
Hard disk	Toshiba SSD 512 G
System	Ubuntu 16.04

5.1. Experimental environments

The experiments are conducted on a laboratory computer. Its configuration is shown in Table 1. The operating system has Ubuntu 16.04 installed. The principally required packages include python 2.7, CUDA8.0, cuDNN7, Tensorflow1.1.0, Caffe, Keras1.2.0, MXNet0.12.0 and others.

5.2. Dataset description

We validate our method on two benchmarks of aerial image labelling, Vaihingen and Potsdam, provided by Commission II/4 of the ISPRS. We conduct experimental evaluations on both the ISPRS Vaihingen and Potsdam 2D semantic labelling challenges. Both are open benchmark datasets provided online and are remote sensing research datasets that describe the environment and surroundings in and around the cities. They include the six most common land cover classes, impervious surfaces (imp_surf), buildings (building), low vegetation (low_veg), trees (tree), cars (car) and clutter/background (clutter).

5.2.1. ISPRS Vaihingen challenge dataset

The Vaihingen dataset contains 33 orthorectified patches (of different sizes) acquired using a near-infrared - green (G) - red (R) aerial camera over the town of Vaihingen (Germany). Images are accompanied by a DSM representing the absolute heights of pixels. We use both the IRRG and DSM images for training.

The average size of the tiles is 2, 494 × 2, 064 pixels with a spatial resolution of 9 cm. Recently, the challenge organiser opened the ground truths of all the images. Among previous opened ground truths, we use 11 annotated images to train the networks and 5 images (ID 11, 15, 28, 30 and 34) to validate training. We use 17 patches as a test set to evaluate the segmentation generalisation accuracy.

5.2.2. ISPRS Potsdam challenge dataset

The Potsdam dataset contains 38 orthorectified same-size patches of size 6000 × 6000 pixels with a spatial resolution of 5 cm over the town of Potsdam (Germany). This dataset offers near-infrared, red, green and blue channels together with the DSM and normalised DSM (NDSM). We use the RGB, IRRG and DSM images for training. From the available patches, 18 densely annotated tiles comprise the training set and 5 tiles (ID 7_7, 7_8, 7_9, 7_11 and 7_12) comprise the validation set, with the same classes as for the Vaihingen dataset. We use only 17 tiles to train the networks. The image numbered 7_10 has error annotations, which can degrade network segmentation performance. It was removed in the following experiments.

5.3. Evaluation metrics

The performance of TreeUNet is evaluated on the ISPRS 2D dataset for OA. To evaluate class-specific performance, the F1 metric is used, computed as the harmonic mean between precision and recall. We also include the mean F1 measure among classes, since OA tends to be less sensitive to minority classes in imbalanced datasets. The OA and F1 metrics can be calculated using the following formulas:

$$\mathrm{OA} = \frac{tp + tn}{p + n}$$

$$F1 = 2 \times \frac{precision \times recall}{precision + recall}$$

$$\mathrm{precision} = \frac{\mathrm{tp}}{\mathrm{tp} + \mathrm{fp}}$$

$$\mathrm{recall} = \frac{\mathrm{tp}}{\mathrm{tp} + \mathrm{fn}}$$

In the pixel-based confusion matrix (ISPRS, 2016), the true positive (tp) is the value of the corresponding diagonal elements. The false positive (fp) is computed from the summation of the column, whereas the false negative (fn) is the summation of the row, excluding the main diagonal element.

5.4. Comparing methods

To verify the performance, the proposed TreeUNet is compared with state-of-the-art methods in two respects: benchmark test comparison and deep model comparison.

5.4.1. Benchmark comparison methods

Submitting the results of the test set to the ISPRS committee enabled us to compare TreeUNet with other competitors on the website, listed as follows:

(1) **TreeUNet**: In our method, only raw image data are used for training. Specifically, 4-band IR/R/G/DSM images are used for Vaihingen and 5-band IR/R/G/B/DSM images are used for Potsdam. Moreover, we do not use any feature fusion techniques, additional hand-crafted features, model ensemble strategy or postprocessing.

(2) **SVL-features + DSM + Boosting + CRF ('SVL_3')**: This was the method implemented as the baseline by the challenge organiser (Gerke, 2015). The normalised digital vegetation index (NDVI), saturation and NDSM features are used in addition to the standard SVL-features (Gould et al., 2011). Then, an Adaboost-based classifier is trained, and a CRF model is applied to obtain a final prediction.

(3) **CNN + NDSM + Deconvolution ('UZ_1')**: This is the method proposed by Volpi and Tuia (2017). They use an encoder–decoder architecture in which rough spatial maps are first learned by convolutions and then upsampled by deconvolution. NDSM data are used in their method.

(4) **CNN + DSM + NDSM + RF + CRF ('ADL_3')**: This is the method proposed by Paisitkriangkrai et al. (2016). They apply both CNN and hand-crafted features to dense image patches to produce per-pixel category probabilities. A random forest (RF) classifier is trained on hand-crafted features, and the output probabilities are combined with those generated by the CNN. CRF is applied as a postprocessing step.

(5) **FCN + DSM + RF + CRF ('DST_2'):** This is the method proposed in Sherrah (2016). This method uses a hybrid FCN architecture to combine image data with DSM data. Then, CRF is applied as a post-processing step.

(6) **FCN + SegNet + VGG + DSM + Edge ('DLR_8')**: This is the method proposed by Marmanis et al. (2018). They use a multiscale ensemble of FCN, SegNet and VGG, incorporating both image data and DSM data. Moreover, they combine semantic labelling with informed edge detection.

(7) **SegNet + DSM + NDSM ('ONE_7')**: This is the method proposed by Audebert et al. (2017). They fuse the output of two multiscale SegNets, which are each then trained with IRRG images and synthetic data (NDVI, DSM and NDSM), respectively.

(8) **ResNet + pretrain + cascade ('CASIA2')**: This method was proposed by Liu et al. (2017). They aggregate the multiscale context using a self-cascaded CNN and the ResNet101 pretrained model and tune the weights with raw input image. They do not use the elevation data (DSM and NDSM) or any postprocessing.

(9) **ResNet + pretrain (SWJ_2)**: This method is not fully published and is described only in a brief abstract. This method uses ResNet101 to

tune a supervised semantic segmentation model and fuse multiscale features using an adaptive approach. It uses only IRRG images for training and testing.

(10) **FCN8s + ResNet + pretrain + ensemble (BKHN10):** This method is not fully published and is described only in a brief abstract. This method uses ensemble multiple models including FCN8s and pre-trained ResNet101s. It uses IRRG, DSM and nDSM images for training and testing.

5.4.2. Comparing deep models

TreeUNet is compared with the following deep models under the same experimental settings. These models have been implemented by our group and can be found on its GitHub website.

(1) **Ours-DeepLab v3+**: This is TreeUNet* with its segmentation model based on DeepLab v3+ (Chen et al., 2018b). DeepLab v3+ has only one Xception in the middle of the flow.
(2) **Ours-DeepUNet**: This is TreeUNet with its segmentation model based on DeepUNet (Li et al., 2018).
(3) **DeepUNet**: This was proposed by Li et al. (2018) for semantic segmentation on VHR remote sensing images. Because it uses residual skip connections, DeepUNet is superior to other VGG-based encoder–decoder architectures for VHR remote sensing image segmentation.
(4) **DeepLab v3+**: This extends DeepLab v3 (Chen et al., 2018b) by adding a simple but effective decoder module to refine the segmentation results especially along object boundaries. It attains a new state-of-the-art performance on the PASCAL VOC 2012 and Cityscapes datasets.

5.5. Results and analysis

5.5.1. Comparison on benchmark sets

To evaluate the effectiveness of the proposed TreeUNet, comparisons with competitors' methods on the two challenging benchmarks are presented.

(1). Vaihingen challenge:

Table 2 shows the quantitative comparison results on the Vaihingen challenge, and Fig. 9 shows the qualitative comparisons. The proposed TreeUNet gets an OA of 90.4%, surpassing the other listed competitors except CASIA2 and BKHN10. Differently from TreeUNet, CASIA2 and BKHN10 both use ResNet101, a much deeper and more difficult to train network, as their primary infrastructure. To avoid overfitting, pre-trained models on ImageNet are adapted to their tasks. We did not use pre-trained models in our experiments. The input of TreeUNet consists of images with five channels (IR/R/G/B/DSM); there is no pre-trained model on a five-channel image dataset. Without pre-trained models and model fusion, TreeUNet assists in distinguishing confusing categories and obtains a more balanced F1 score on each category. As observed, even the OA of CASIA2 is higher than ours, and the Mean F1 of TreeUNet is very close to that of CASIA2. Fig. 10 shows a qualitative comparison on three sampled patches. Compared with ADL_3, DLR_10, DST_2, UZ_1 and ONE_7, the TreeUNet results have sharper and clearer boundaries on buildings.

(2). Potsdam challenge:

Table 3 shows the quantitative comparisons on the Potsdam images, and Fig. 10 shows the qualitative comparisons. The OA of TreeUNet is also listed in third place. Just as in the previous analyses on Vaihingen, the fact that SWJ_2 and CASIA2 surpass TreeUNet is most likely a result of using deeper neural networks and pre-trained models. However, without any pretrained models or other complex fusion techniques, our results are competitive. In Fig. 10, TreeUNet shows better performance on cluttered areas and is less influenced by shadows. For tiny objects like cars, our results are still satisfactory in reducing the adhesion of pixels and the bubble effect.

5.5.2. Comparison with deep models

To evaluate the effectiveness of the proposed TreeUNet architecture, comparisons with general deep semantic models are presented. In this group of experiments, we used the same training data, data augmentation methods and learning strategy to see how different deep network structures influence performance.

(1) Vaihingen test set

Table 4 shows the quantitative comparisons between TreeUNet and the deep models without a Tree-CNN block on the ISPRS Vaihingen test set. TreeUNet is based on DeepUNet, whereas TreeUNet* is based on DeepLab v3+. As for OA, TreeUNet scores are 1.2% higher than DeepUNet, and TreeUNet* scores are 0.4% higher than DeepLab v3+. The adaptive Tree-CNN block combined with the concatenating connections enhances the OA of pixel-wise semantic segmentation as well as the F1 metrics on easily confused categories.

On the Vaihingen dataset, we found two pairs of easily confused categories: (1) imp_surf and clutter and (2) low_veg and tree. With regard to the categories of imp_surf and clutter, the F1 scores of TreeUNet increased by 1.0% and 4.9% compared with those of the original DeepUNet, whereas those of TreeUNet* increased by 0.7% and 6.7% compared with those of DeepLab v3+. With regard to another pair of easily confused categories, low_veg and tree, TreeUNet scores are 0.9% and 0.6% higher than DeepUNet, whereas TreeUNet* scores are 0.6% and 0.2% higher than DeepLab v3+.

(2) Potsdam test set

Table 5 shows the quantitative comparisons between TreeUNet and the deep models without a Tree-CNN block on the ISPRS Potsdam test set. Similar results are obtained. TreeUNet scores are better than DeepUNet scores, and TreeUNet* scores are better than DeepLab v3+ scores. The use of an adaptive Tree-CNN block improves the performance significantly. The OA of TreeUNet is improved by 1.7% compared with that of DeepUNet, whereas that of TreeUNet* is improved

Table 2
Quantitative comparisons between TreeUNet and other published methods on the ISPRS Vaihingen test set.

Method	Imp_surf	Building	Low_veg	Tree	Car	Clutter	OA	Mean F1
SVL_3	86.6	91.0	77.0	85.0	55.6	–	84.8	79.0
DST_2	90.5	93.7	83.4	89.2	72.6	–	89.1	85.9
UZ_1	89.2	92.5	81.6	86.9	57.3	–	87.3	81.5
RIT_L7	90.1	93.2	81.4	87.2	72.0	–	87.8	84.8
ONE_7	91.0	94.5	84.4	89.9	77.8	–	89.8	87.5
ADL_3	89.5	93.2	82.3	88.2	63.3	–	88.0	83.3
DLR_10	92.3	95.2	84.1	90.0	79.3	–	90.3	88.2
CASIA2	93.2	96.0	84.7	89.9	86.7	–	91.1	90.1
BKHN10	92.9	96.0	84.6	89.8	88.8		91.0	90.4
TreeUNet	92.5	94.9	83.6	89.6	85.9	–	90.4	**89.3**

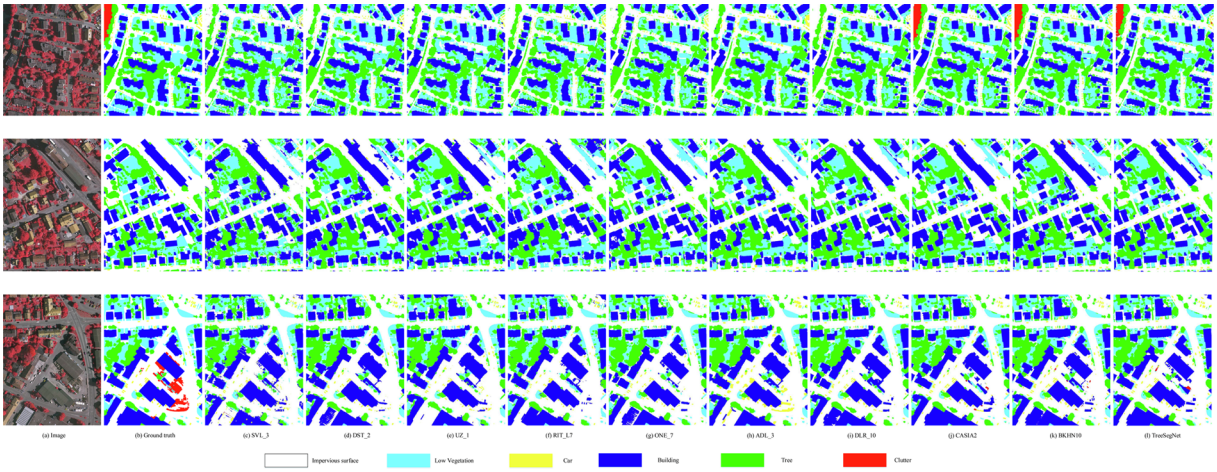

Fig. 9. Comparisons between TreeUNet and other published methods on the ISPRS Vaihingen test set.

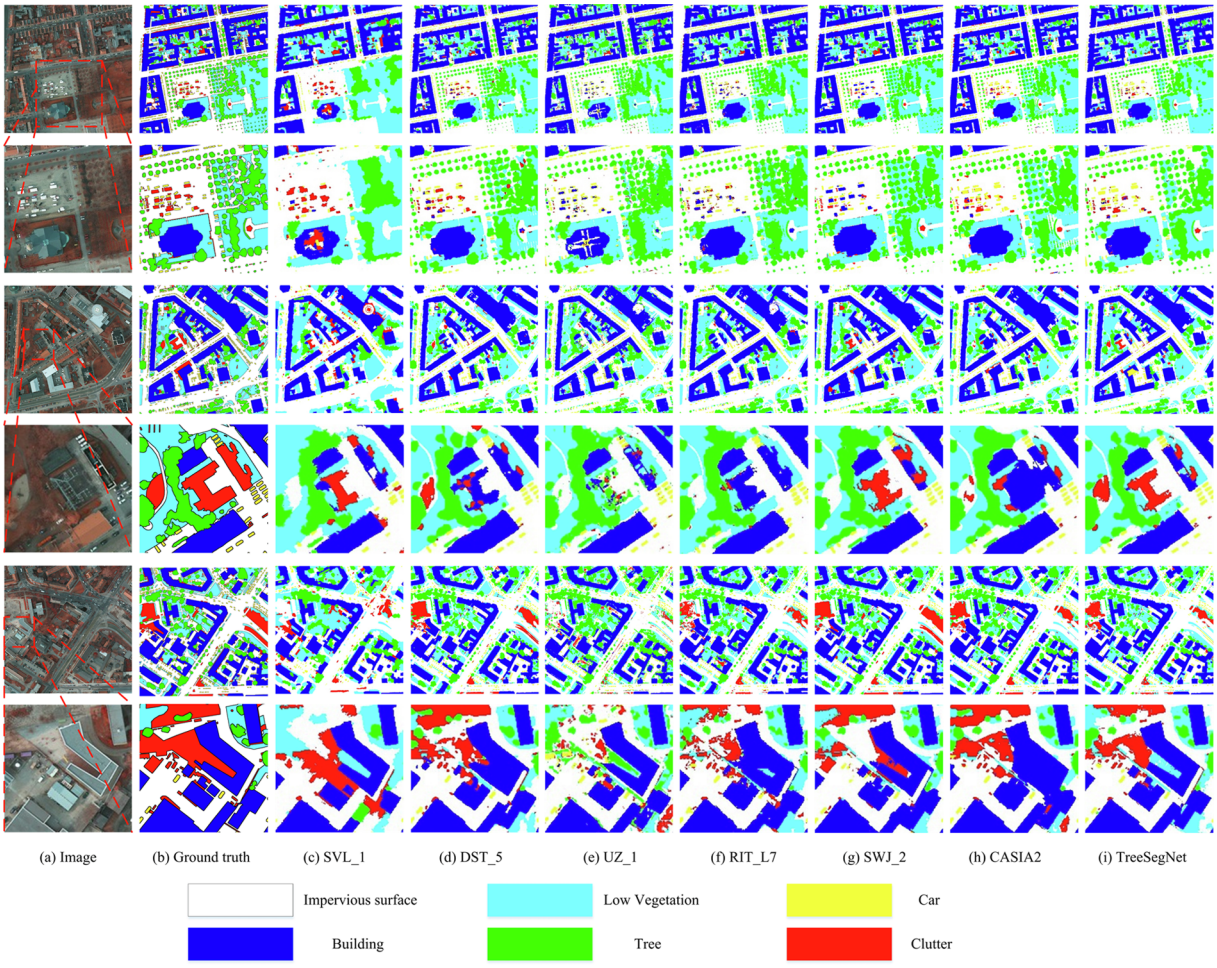

Fig. 10. Comparisons between TreeUNet and other published methods on the ISPRS Potsdam test set.

by 1.3%. The TreeUNet architecture is not inconsistent with the state-of-the-art deep models. The problem of easily confused categories can be alleviated by replacing the segmentation module with the latest deep model, thereby improving the performance further.

VHR remote sensing images contain more contextual information, thereby requiring deeper and more powerful models for feature extraction. DeepLab v3+ contains more convolutional layers and is therefore thought to be able to understand context better. However, the experiments produced the opposite results. We attempted to analyse this phenomenon by increasing the number of Xception blocks in the middle of the flow in DeepLab v3+. As a result, the generalisation performance decreased. This shows that the training data are not sufficient for the deep neural network. We expect that using pre-trained models or decreasing the number of neural layers can alleviate the situation.

For the next several groups of experiments from Sections 5.5.3 to 5.5.5, we divide the labelled Potsdam datasets into two parts: 18 images as the training set and 5 images (image numbers 7_7, 7_8, 7_9, 7_11 and

Table 3
Quantitative comparisons between TreeUNet and other published methods on the ISPRS Potsdam test set.

Method	Imp_surf	Building	Low_veg	Tree	Car	OA	Mean F1
SVL_1	83.5	91.7	72.2	63.2	62.2	77.8	74.6
DST_5	92.5	96.4	86.7	88.0	94.7	90.3	91.7
UZ_1	89.3	95.4	81.8	80.5	86.5	85.8	86.7
RIT_L7	91.2	94.6	85.1	85.1	92.8	88.4	89.8
SWJ_2	94.4	97.4	87.8	87.6	94.7	91.7	92.4
CASIA2	93.3	97.0	87.7	88.4	96.2	91.1	92.5
TreeUNet	93.1	97.3	86.8	87.1	95.8	90.7	92.0

7_12) as the validation set. The following results are reported on the validation set unless otherwise specified.

5.5.3. Different tree structures

To understand the benefit provided by the Tree-CNN structures, we show the results of a detailed analysis of the Potsdam dataset in Fig. 11 and Table 6. This set of experiments used 64 concatenating features tested through TreeUNet (DeepUNet). Two common methods for increasing the complexity of the network structure and achieving better performance are to widen or deepen the network. The Tree-CNN structures can be chosen in many ways, for example, by using a binary balanced tree structure (widen the network) or by increasing the convolutional layers directly by adding a 'straight tree' structure (deepen the network). Thus, it is necessary to prove that the higher OA of TreeUNet is not a result of having a redundant deeper or wider network structure. We designed two special tree structures, the elegant balanced tree shown in Fig. 11a and the straight tree structure shown in Fig. 11b, and we trained them under the same conditions. In contrast, TreeUNet is constructed dynamically and adaptively. For the first iteration, it is trained without the Tree-CNN block. Then, the Tree-CNN block is constructed from the confusion matrix derived from the initial segmentation results shown in Fig. 11c. Next, the Tree-CNN block is updated according the same procedure to the diagram shown in Fig. 11d. These steps are iterated until the structure of the Tree-CNN block no longer changes. The network we propose is based on multi-stage training. Each pass is a phase of network training including 80 epochs. During the training of each pass, the structure of the Tree-CNN block remains stable. According to the experimental results, the evolution of the confusion trees changes from none to Fig. 11c and then to those in Fig. 11d. Table 6 shows the comparison results. TreeUNet with the iterated confusion tree structure shown in Fig. 11d obtained the highest OA of 90.66%.

Our experiments show that the Tree-CNN block adaptively learned through network segmentation results performs best on the segmentation results in contrast to artificially designed ones. As the structure of the Tree-CNN block updates, the OA of the network segmentation results improves step by step.

5.5.4. Different concatenating features

The number of features on the concatenating connection is a key parameter. We tested different numbers of features in the concatenating connections, using 16, 32 and 64 features. The OA for three different feature numbers on Potsdam 2D is shown in Fig. 12. Because the highest OA was 90.6%, we selected 64 as the number of features in the concatenating connections. The tree structure used is similar to that in Fig. 11d.

Table 6
The OA scores of the validation set for different structures of TreeUNet (DeepUNet) on Potsdam 2D.

Structure	OA
TreeUNet with no tree	88.43
TreeUNet + balanced tree (Fig. 12. a)	89.88
TreeUNet + straight structure (Fig. 12. b)	89.22
TreeUNet + 1 pass (Fig. 12. c)	90.61
TreeUNet + 2 passes (Fig. 12. d)	**90.66**

Table 4
Quantitative comparisons between TreeUNet and other deep models on the ISPRS Vaihingen test set. (The Mean F1 score does not use the data of the clutter class.)

Method	Imp_surf	Building	Low_veg	Tree	Car	Clutter	OA	Mean F1
DeepLab v3+	88.4	91.9	77.6	85.2	68.3	31.5	85.9	82.3
TreeUNet*	89.1	92.5	78.2	85.4	74.0	38.2	86.3	83.9
DeepUNet	91.5	94.1	82.7	89.3	85.5	47.1	89.3	88.7
TreeUNet	92.5	94.9	83.6	89.6	85.9	52.0	90.4	89.3

Table 5
Quantitative comparisons between TreeUNet and other deep models on the ISPRS Potsdam test set. (The Mean F1 score does not use the data of the clutter class.)

Method	Imp_surf	Building	Low_veg	Tree	Car	Clutter	OA	Mean F1
DeepLab v3+	90.3	95.8	83.8	81.0	93.7	39.9	87.4	87.4
TreeUNet*	90.7	97.1	86.4	85.5	93.9	41.2	89.1	90.6
DeepUNet	92.8	97.4	84.5	85.5	95.1	44.7	89.4	91.1
TreeUNet	93.1	97.3	86.8	87.1	95.8	53.0	90.7	92.0

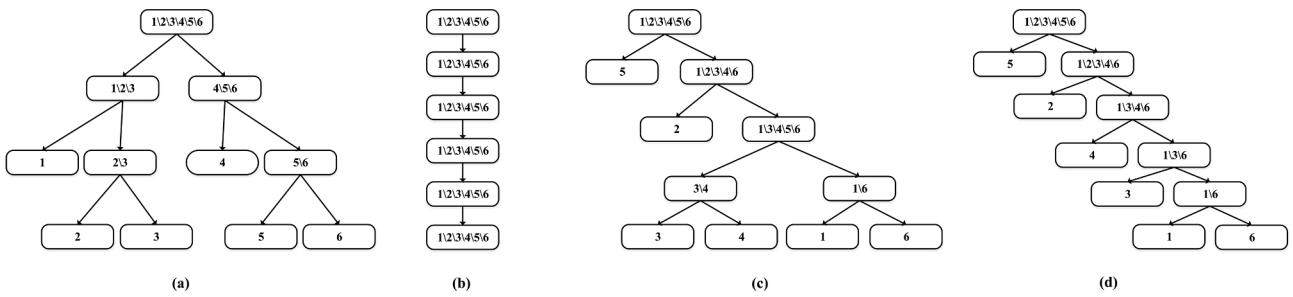

Fig. 11. The different tree structures in Table 6.

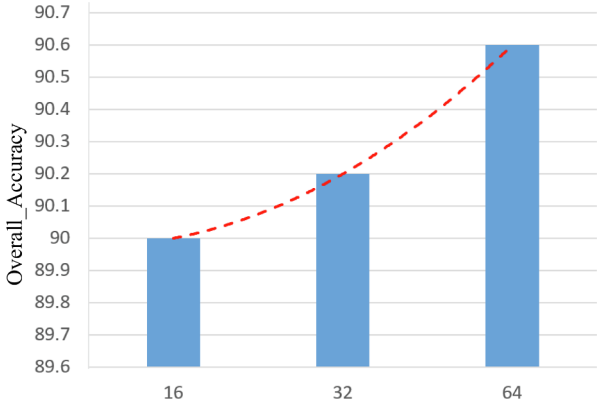

Fig. 12. Influence of the number of features in the concatenating connections on the OA. It was tested using TreeUNet (DeepUNet) on Potsdam 2D.

6. Conclusions

In this paper, we proposed a new approach to addressing the semantic segmentation of VHR remote sensing images, using TreeUNet with an automatic constructed Tree-CNN block. In the experiments on both the ISPRS Vaihingen and Potsdam datasets, the F1 scores of the easily confused categories are all improved. Finally, we obtained the competitive OA values among state-of-the-art methods.

In the future, other data augmentation methods could be considered, for example, using generative adversarial networks. With the augmented data, we could try deeper neural networks and endeavour to design neural networks on different domains.

Acknowledgements

The authors would like to acknowledge the provision of the datasets by ISPRS and BSF Swiss photo, which were released in conjunction with the ISPRS, led by ISPRS WG II/4.

The authors wish to thank the editors and anonymous reviewers for their valuable comments which greatly improved the paper's quality.

My paper is supported by the Project supported by the National Natural Science Foundation of China (General Program; Key Program; Major Program). Grant No 61871413.

Appendix A

The recursive TreeCutting algorithm

The undirected graph is then transformed into a binary tree structure using the proposed TreeCutting algorithm. Pseudocode for the algorithm is shown in Algorithm 1. It is a recursive function. Cutting of an edge with the minimum weight is executed iteratively. After each iteration, the algorithm checks to determine whether the current graph (G) is divided into two disconnected subgraphs. If so, it creates a node (T) and performs TreeCutting on the subgraphs (G_1 and G_2) recursively; otherwise, it continues to the next iteration. The TreeCutting algorithm is different from the classic min-cut/max-flow algorithm in two respects. First, their objectives are different. TreeCutting aims to greedily construct a binary tree fast, whereas the min-cut/max-flow algorithm attempts to find the minimum set of edges to partition the connected graph optimally. Second, the complexities of the algorithms are different. Assuming N is the number of nodes, the TreeCutting algorithm has $O(n^2 \log n)$ complexity, whereas the min-cut/max-flow algorithm has $O(n^4)$ complexity, which takes a longer time when N is large. Even the Stoer–Wanger minimum cut algorithm requires $O(n^3)$ running time. Actually, to construct the tree, either TreeCutting or Stoer–Wanger could be chosen if N is small. The comparisons described above are intended only for the complexities of the two algorithms. For these small graphs, the time difference between the two algorithms in terms of complexity can be ignored compared with the time for training the network.

Algorithm 1. The algorithm for the proposed **TreeCutting** operation

Input: The undirected confusion graph, $G(V, E)$
The Point set of graph G, V
The Edge set of graph$G$, E
Output: The binary tree-like structure T
while E still has edges ($E \neq \emptyset$) **do**
select $e \in E$ with minimum weight
$G \leftarrow G(V, E - e)$
if G is still a connected graph **then**
continue
elif G is cut into two subgraphs $G_1(V_1, E_1)$ & $G_2(V_2, E_2)$ **then**
$T.\ leftchild \leftarrow V_1$
$T.\ rightchild \leftarrow V_2$
TreeCutting(G_1, V_1, E_1)
TreeCutting(G_2, V_2, E_2)
end if
end while

Appendix B

Different tree building algorithms

We compare the trees constructed using the TreeCutting algorithm with those constructed using the min-cut/max-flow algorithm in the experiments. We use the Stoer–Wagner minimum cut algorithm for undirected graphs. On the Vaihingen and Potsdam datasets, we obtained four confusion matrices before convergence, as Fig. 13 shows. It was found that trees constructed using TreeCutting and those constructed using the minimum-cut algorithm are the same even though the procedures for constructing them are different. To further understand the problem, we

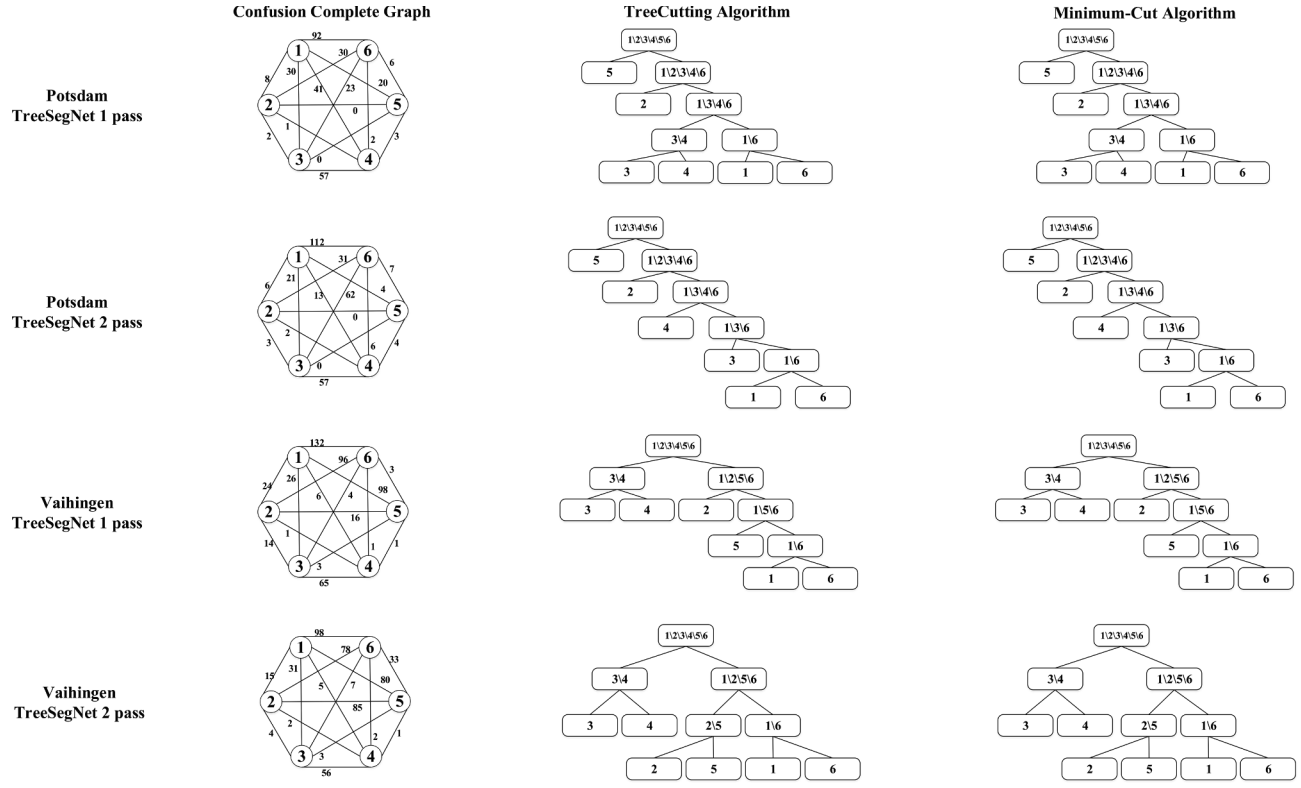

Fig. 13. Comparison between the TreeCutting and minimum-cut algorithms tested by TreeUNet (DeepUNet) on Potsdam 2D.

changed the weights of the confusion matrices in the experiments randomly with the two algorithms always obtaining the same results. In fact, the minimum-cut algorithm provides an optimised means of building the tree structure, whereas TreeCutting can be seen as a fast approximation of the minimum-cut algorithm for undirected complete graphs.

References

Audebert, N., Le Saux, B., Lefèvre, S., 2018. Beyond RGB: very high resolution urban remote sensing with multimodal deep networks. ISPRS J. Photogramm. Remote Sens. 140, 20–32. https://doi.org/10.1016/j.isprsjprs.2017.11.011.

Audebert, N., Le Saux, B., Lefèvre, S., 2017. Semantic segmentation of earth observation data using multimodal and multi-scale deep networks. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics).

Badrinarayanan, V., Kendall, A., Cipolla, R., 2017. SegNet: a deep convolutional encoder-decoder architecture for image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 39, 2481–2495. https://doi.org/10.1109/TPAMI.2016.2644615.

Bengio, E., Bacon, P.-L., Pineau, J., Precup, D., 2015. Conditional Computation in Neural Networks for faster models. arXiv Prepr. arXiv1511.06297, 2015.

Boykov, Y.Y., 2001. Interactive graph cuts for optimal boundary & region segmentation of objects in ND images. In: Comput. Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE Int. Conf., vol. 1. pp. 105–112.

Chen, L.-C., Papandreou, G., Schroff, F., Adam, H., 2017. Rethinking Atrous Convolution for Semantic Image Segmentation. arXiv Prepr. arXiv1706.05587, 2017.

Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L., 2018a. DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. IEEE Trans. Pattern Anal. Mach. Intell. 40, 834–848. https://doi.org/10.1109/TPAMI.2017.2699184.

Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H., 2018b. Encoder-decoder with atrous separable convolution for semantic image segmentation. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics).

Cui, W., Zhang, Y., 2011. An effective graph-based hierarchy image segmentation. Intell. Autom. Soft Comput. 17, 969–981. https://doi.org/10.1080/10798587.2011.10643203.

Deng, J., Satheesh, S., Berg, A.C., et al., 2011. Fast and balanced: Efficient label tree learning for large scale object recognition. Adv. Neural Inf. Process. Syst. 567–575.

Di, Y., Jiang, G., Yan, L., Liu, H., Zheng, S., 2017. Multi-scale segmentation of high resolution remote sensing images by integrating multiple features. ISPRS – Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. XLII-1/W1 247–255. https://doi.org/10.5194/isprs-archives-XLII-1-W1-247-2017.

Figurnov, M., Collins, M.D., Zhu, Y., Zhang, L., Huang, J., Vetrov, D., Salakhutdinov, R., 2017. Spatially adaptive computation time for residual networks. In: Proceedings – 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, pp. 1790–1799.

Gao, T., Koller, D., 2011. Discriminative learning of relaxed hierarchy for large-scale visual recognition. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2072–2079.

Gerke, M., 2015. Use of the stair vision library within the ISPRS 2D semantic labeling benchmark (Vaihingen). Tech. Report.

Gould, S., Russakovsky, O., Goodfellow, I., Baumstarck, P., 2011. The Stair Vision Library (v2.5). Stanford Univ.

Grady, L., 2006. Random walks for image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 28, 1768–1783. https://doi.org/10.1109/TPAMI.2006.233.

Grady, L., Schwartz, E.L., 2004. Isoperimetric graph partitioning for data clustering. IEEE Pattern Anal. Mach. Intell. XX 1–30.

Griffin, G., Perona, P., 2008. Learning and using taxonomies for fast visual categorization. 26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR.

ISPRS, 2016. International Society for Photogrammetry and Remote Sensing. 2D Semantic Labeling Challenge. [WWW Document]. URL http://www2.isprs.org/commissions/comm3/wg4/semantic-labeling.html (accessed 12.21.18).

Krizhevsky, A., Sutskever, I., Geoffrey, E.H., 2012. ImageNet classification with deep convolutional neural networks. Adv. Neural Inf. Process. Syst. 25, 1–9. https://doi.org/10.1109/5.726791.

Leroux, S., Bohez, S., De Coninck, E., Verbelen, T., Vankeirsbilck, B., Simoens, P., Dhoedt, B., 2017. The cascading neural network: building the Internet of Smart Things. Inf. Syst. Knowl. https://doi.org/10.1007/s10115-017-1029-1.

Li, R., Liu, W., Yang, L., Sun, S., Hu, W., Zhang, F., Li, W., 2018. DeepUNet: a deep fully convolutional network for pixel-level sea-land segmentation. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. https://doi.org/10.1109/JSTARS.2018.2833382.

Li, Z., Hoiem, D., 2016. Learning without forgetting. In: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pp. 614–629.

Liu, Y., Piramanayagam, S., Monteiro, S.T., Saber, E., 2017. Dense semantic labeling of very-high-resolution aerial imagery and LiDAR with fully-convolutional neural networks and higher-order CRFs. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 1561–1570.

Long, J., Shelhamer, E., Darrell, T., 2015. Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 3431–3440.

Maggiori, E., Tarabalka, Y., Charpiat, G., Alliez, P., 2017. High-resolution aerial image labeling with convolutional neural networks. IEEE Trans. Geosci. Remote Sens.

https://doi.org/10.1109/TGRS.2017.2740362.
Marmanis, D., Schindler, K., Wegner, J.D., Galliani, S., Datcu, M., Stilla, U., 2018. Classification with an edge: improving semantic image segmentation with boundary detection. ISPRS J. Photogramm. Remote Sens. https://doi.org/10.1016/j.isprsjprs.2017.11.009.
Marszałek, M., Schmid, C., 2008. Constructing category hierarchies for visual recognition. In: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pp. 479–491.
Noh, H., Hong, S., Han, B., 2015. Learning deconvolution network for semantic segmentation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1520–1528.
Odena, A., Lawson, D., Olah, C., 2017. Changing Model Behavior at Test-Time Using Reinforcement Learning. arXiv Prepr. arXiv1702.07780, 2017. http://doi.org/10.1051/0004-6361/201527329.
Ouma, Y.O., Tetuko, J., Tateishi, R., 2008. Analysis of co-occurrence and discrete wavelet transform textures for differentiation of forest and non-forest vegetation in very-high-resolution optical-sensor imagery. Int. J. Remote Sens. 29, 3417–3456. https://doi.org/10.1080/01431160701601782.
Paisitkriangkrai, S., Sherrah, J., Janney, P., Van Den Hengel, A., 2016. Semantic labeling of aerial and satellite imagery. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. https://doi.org/10.1109/JSTARS.2016.2582921.
Reis, S., Taşdemir, K., 2011. Identification of hazelnut fields using spectral and gabor textural features. ISPRS J. Photogramm. Remote Sens. 66, 652–661. https://doi.org/10.1016/j.isprsjprs.2011.04.006.
Ronneberger, O., Fischer, P., Brox, T., 2015. U-net: Convolutional networks for biomedical image segmentation. In: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pp. 234–241.
Rusu, A.A., Rabinowitz, N.C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., Hadsell, R., 2016. Progressive Neural Networks. arXiv Prepr. arXiv1606.04671, 2016.
Sherrah, J., 2016. Fully Convolutional Networks for Dense Semantic Labelling of High-Resolution Aerial Imagery. arXiv Prepr. arXiv1606.02585, 2016.
Shi, J., Malik, J., 2000. Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 22, 888–905. https://doi.org/10.1109/34.868688.
Simonyan, K., Zisserman, A., 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv Prepr. arXiv1409.1556, 2014. http://doi.org/10.1016/j.infsof.2008.09.005.
Srivastava, N., Salakhutdinov, R.R., 2013. Discriminative transfer learning with tree-based priors. Adv. Neural Inf. Process. Syst. 2, 1–9. https://doi.org/10.1017/S0272263100006331.
Tanno R, Arulkumaran K, Alexander D C, et al., 2018. Adaptive Neural Trees. arXiv Prepr. arXiv1807.06699, 2018.
Volpi, M., Tuia, D., 2018. Deep multi-task learning for a geographically-regularized semantic segmentation of aerial images. ISPRS J. Photogramm. Remote Sens. https://doi.org/10.1016/j.isprsjprs.2018.06.007.
Volpi, M., Tuia, D., 2017. Dense semantic labeling of subdecimeter resolution images with convolutional neural networks. IEEE Trans. Geosci. Remote Sens. 55, 881–893. https://doi.org/10.1109/TGRS.2016.2616585.
Wang, H., Wang, Y., Zhang, Q., Xiang, S., Pan, C., 2017. Gated convolutional neural network for semantic segmentation in high-resolution images. Remote Sens. 9. https://doi.org/10.3390/rs9050446.
Wang, T., Zhang, H., Lin, H., Fang, C., 2016. Textural-spectral feature-based species classification of mangroves in Mai Po nature reserve from worldview-3 imagery. Remote Sens. 8. https://doi.org/10.3390/rs8010024.
Wilkinson, M., Bulloch, B., Garcia-Filion, P.A.M., Keahey, L., 2011. Efficacy of racemic albuterol versus levalbuterol used as a continuous nebulization for the treatment of acute asthma exacerbations: A randomized, double-blind, clinical trial. J. Asthma 48, 188–193. https://doi.org/10.3109/02770903.2011.554939.
Xiao, T., Zhang, J., Yang, K., Peng, Y., Zhang, Z., 2014. Error-driven incremental learning in deep convolutional neural network for large-scale image classification. In: Proceedings of the ACM International Conference on Multimedia – MM '14, pp. 177–186.
Yan, Z., Zhang, H., Piramuthu, R., Jagadeesh, V., Decoste, D., Di, W., Yu, Y., 2015. HD-CNN: Hierarchical deep convolutional neural networks for large scale visual recognition. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2740–2748. https://doi.org/10.1109/ICCV.2015.314.
Ye, W., Wang, Y., 2009. MST image segmentation based on Mumford-Shah theory. J. Comput. Des. Comput. Graph. 21, 014.
Yoon, J., Yang, E., Lee, J., Hwang, S.J., 2017. Lifelong Learning with Dynamically Expandable Networks. CoRR, 2017.
Yu, H., Yang, W., Xia, G.-S., Liu, G., 2016. A color-texture-structure descriptor for high-resolution satellite image classification. Remote Sens. 8, 259. https://doi.org/10.3390/rs8030259.