Road Segmentation for Remote Sensing Images Using Adversarial Spatial Pyramid Networks

Pourya Shamsolmoali, *Member, IEEE*, Masoumeh Zareapoor, Huiyu Zhou, Ruili Wang, and Jie Yang

***Abstract*— Road extraction in remote sensing images is of great importance for a wide range of applications. Because of the complex background, and high density, most of the existing methods fail to accurately extract a road network that appears correct and complete. Moreover, they suffer from either insufficient training data or high costs of manual annotation. To address these problems, we introduce a new model to apply structured domain adaption for synthetic image generation and road segmentation. We incorporate a feature pyramid (FP) network into generative adversarial networks to minimize the difference between the source and target domains. A generator is learned to produce quality synthetic images, and the discriminator attempts to distinguish them. We also propose a FP network that improves the performance of the proposed model by extracting effective features from all the layers of the network for describing different scales' objects. Indeed, a novel scale-wise architecture is introduced to learn from the multilevel feature maps and improve the semantics of the features. For optimization, the model is trained by a joint reconstruction loss function, which minimizes the difference between the fake images and the real ones. A wide range of experiments on three data sets prove the superior performance of the proposed approach in terms of accuracy and efficiency. In particular, our model achieves state-of-the-art 78.86 IOU on the Massachusetts data set with 14.89M parameters and 86.78B FLOPs, with 4× fewer FLOPs but higher accuracy (+3.47% IOU) than the top performer among state-of-the-art approaches used in the evaluation.**

***Index Terms*— Adversarial network, domain adaptation, feature pyramid (FP), remote sensing (RS) images, road segmentation.**

I. Introduction

CURRENTLY road segmentation in remote sensing (RS) images has become one of the crucial tasks in many urban applications such as traffic management, urban planning, and road monitoring. Meanwhile, it is tremendously time-consuming to manually label roads from the high-resolution images. Unsupervised models, which are based on the predefined features, achieved low accuracy and failed on heterogeneous regions. However, supervised deep learning models, have achieved high performance in most of computer vision tasks, such as object detection [1]–[3], semantic segmentation [4]–[6], and skeleton extraction [7]. With the improvement of convolution neural networks, road detection from RS images tends to be an efficient and effective process.

Actually, the road network detection contains two subtasks: road edge detection and road surface segmentation that meet several problems: semantic segmentation and object extraction. Generally, buildings, cars, and trees along the roads create shadows or can lead to occlusions that cause heterogeneous regions on roads. Hence, road detection is a challenging task. However, the current methods fail to tackle the abovementioned issues, in which the benchmark data sets are usually selected from the urban areas. With the great success of convolutional neural networks and fully connected networks, several architectures have been proposed for RS road detection and segmentation [3], [6]. On the other hand, most of the state-of-the-art road detection methods are fully supervised, demanding massive labeled training data. Data augmentation is a technique that is generally used in the state-of-the-art methods for increasing the number of the training data. Nevertheless, if the distribution between the training and the testing data sets is changed, the performance can still be unsatisfactory. Another approach is using generative adversarial networks (GANs) to produce synthetic images [8]. However, synthetic data are generally more problematic than the realistic data. By using simulation tools, a huge amount of synthetic images for training a neural network can be produced [9], [10]. Nevertheless, the recent works [11], [12] reveal that, still there is a gap between the distribution of the real and that of the synthetic data.

It has been observed that using deep features resulted in minimizing the cross-domain distribution discrepancy, but did not remove it [13]. Transfer learning is a machine learning method in which a developed model for a task is reused for a model on the other task, which is widely used in RS image classification [14]. Transfer learning algorithms, unlike traditional machine learning algorithms, intend to build models that are applicable for different domains or tasks. Domain adaptation is a transfer learning technique, in which the standard domain assumption does not hold, and the learned models from one or more domains may apply for the same task in another domain. In domain adaptation, training is

Manuscript received February 2, 2020; revised June 29, 2020; accepted August 9, 2020. Date of publication August 21, 2020; date of current version May 21, 2021. This work was supported in part by NSFC under Grant 61806125 and Grant 61977046. *(Corresponding author: Jie Yang.)*

Pourya Shamsolmoali, Masoumeh Zareapoor, and Jie Yang are with the Institute of Image Processing and Pattern Recognition, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: jieyang@sjtu.edu.cn).

Huiyu Zhou is with the School of Informatics, University of Leicester, Leicester LE1 7RH, U.K.

Ruili Wang is with the School of Logistics and Transportation, Central South University of Forestry and Technology, Changsha 498, China, and also with the School of Natural and Computational Sciences, Massey University, Auckland 0632, New Zealand.

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TGRS.2020.3016086

conducted in the source domain, and the test samples are taken from the target domain [15]. This article targets at training a road segmentation model for real RS images and parsing them without using any manual annotation. This problem is worthy to be investigated due to three reasons: 1) road detection in RS images has become an attractive theme in both industry and academia, while segmentation is one of the most obligatory procedures in understanding high density and complicated RS images; 2) a large amount of high-resolution and unseen annotated imagery is required to well train a deep neural network for RS road segmentation. In comparison with object detection and classification in RS data, it is an extremely laborious and time-consuming task for a professional to annotate pixel-wise training data for image segmentation. For instance, a synthetic image annotation on average only takes a few seconds. In contrast, single image manual annotation may take more than 1.5 h [16]; and 3) this process requires dedicated equipment and time to gather images which cover a large diverse urban landscape in different areas with different lighting set-ups. Theoretically and practically, it is interesting and worthy to develop a method to conduct automated road segmentation on RS images without manual labeling.

One of the most powerful methods for creating synthetic images is GANs [16]. GANs generally consist of two networks in which the first network, the generator, learns to produce synthesis images that are barely distinguishable from the genuine images. On the other hand, the other network (discriminator) is trained to distinguish the real images from the generated ones. Simultaneously, the generator and the discriminator are trained to minimize the adversarial loss. Most of the present GAN methods generally focus on improving the loss function [17], designing new architectures or tricks [18], or creating new training schemes like the gradual and progressive training [19]. So far, limited works have been reported on the design of neural networks architecture for synthetic image generation [11], [12].

The intuitive idea of this article is to explore new architectures that minimize the feature space gap between the source and target domains, which can improve the performance of adversarial learning for generating realistic synthetic RS images for road segmentation. The network is trained to transform the feature maps of source domain images to those of target domain images while preserving the semantic information of the features. We propose a generic and lightweight network architecture that can easily be integrated into the generator. We also introduce an efficient spatial pyramid network to extract multiscale and global contextual information. The proposed adversarial spatial pyramid network (ASPN) does not require matching image pairs from the corresponding domains that practically do not exist. To the best of our knowledge, this is the first work that integrates spatial feature pyramid (FP) networks with the adversarial domain adaptation for the RS road segmentation. Compared to the existing approaches, the proposed unsupervised domain adaptation architecture has the following advantages.

1) *Network Architecture:* Previous approaches [13], [14] are generally based on two domains through an intermediate feature space and thereby implicitly take the same decision function for both domains. ASPN uses the multiscale feature maps with a conditional generator and residual learning; all the modules of the proposed architecture are placed in a single network and have an end-to-end training strategy. A discriminator is also used [12] to manage adversarial learning.
2) In the generator, we propose an efficient pyramid network architecture to extract effective pyramid features of multiple scales to overcome the current models' limitations. In the proposed pyramid network, first, the multiscale features are extracted and fused as the base features; we next feed them into an optimized U-shape network (OUN) [1] and feature modules concatenation (FMC) to produce additional descriptive and multiscale features. We also use a module called scale-wise feature concatenate (SFC) which is similar to an inception network [5] to merge the features in a wide range of scales. Finally, the feature maps with the same size are collected to form the last FP. The code will be made available on `https://github.com/pshams55/ASPN`.

II. Related Work

Here, we discuss recent works on road segmentation, domain adaptation, and FP networks.

A. Road Segmentation

The urban development led to significant growth of transportation networks. Such infrastructure requires frequent updates of road maps. Previous road detection and segmentation models were based on local features handcrafted by domain experts [20]. Liu *et al.* [7] proposed a multitask pixel wise end-to-end CNN, to simultaneously predict road surfaces, edges, and centerlines. The model learns multiscale and multilevel features and is trained in a specially designed cascaded network, which can deal with the roads in various scenes and scales that deals with several issues in road detection. Henry *et al.* [3] evaluated different FCNs for road segmentation in high-resolution synthetic aperture radar satellite images. Cheng *et al.* [21] proposed an end-to-end cascaded convolutional network for road detection. The model consists of two networks. One aims at the road detection task and the other is cascaded to the former one, achieving full use of the feature maps, to obtain the good centerline extraction. Wei *et al.* [22] proposed a multistage framework to accurately extract the road surface and centerline simultaneously. This framework consists of two FCNs to boost the accuracy and connectivity of the image segmentation.

B. Domain Adaptation and Adversarial Networks

DNN models are highly dependent on the training and testing data that have equal underlying distribution. Practically, it is common to have some diversity between the training and testing data. To rectify this difference and adjust the methods for better generalization, domain adaptation is recommended [13], [14]. Hoffman *et al.* [23] proposed a cycle-consistent domain adaptation model that receives multiple forms of

representations while enforcing local and global structural consistency through pixel cycle-consistency and semantic losses. The model uses a reconstruction loss to boost the cross-domain conversion to maintain structural information and a semantic loss to sustain semantic consistency. Zhu *et al.* [24] proposed a learning method to translate an image from the source domain to the target domain by introducing cycle learning without paired examples. Wang *et al.* [25] proposed a weakly supervised adversarial domain adaptation method to improve the segmentation performance from the synthetic data, which consists of three networks. The detection model focuses on detecting objects and predicting a segmentation map; the pixel-level domain classifier attempts to distinguish the domains of image features; an object-level domain classifier discriminates the objects and predicts the objects classes. Li *et al.* [26] proposed a road segmentation model that combined the adversarial networks with multiscale context aggregation. In this approach, standard U-Net is used as the generator, and FCNs are used as the discriminator. This model is computationally inefficient and has limited performance in handling complex images. Wang *et al.* [27] proposed a new linear space embedded clustering method, which uses adaptive neighbors to optimize the similarity matrix and clustering results simultaneously, also a linearity regularization is used to generate a linear embedded spectral from the data representation.

C. Feature Pyramid Network

For different tasks in high spatial resolution RS images, it has been proposed to use multiscale images for training. FPs are a module in recognition systems for detecting and segmenting of objects with different scales. Plenty of efforts have been made to improve performance accuracy. Bellens *et al.* [28] introduced a bottom-up model for RS image classification. To avoid the indirect classification and segmentation problems of these bottom-up methods, Sivic and Zisseman [29] proposed to recognize objects without classifying pixels or regions. Afterward, a spatial pyramid network is used for instance segmentation [30], [31]. Recently, several FP architectures have been proposed that achieved encouraging results [4], [32], and still the current approaches have limitations as they just build the FPs based on the original pyramidal architecture of the backbones with several scales. For instance, the authors in [2] present a FP architecture for object detection in RS images with several dense blocks and pooling layers. Lin *et al.* [33] presented a FP architecture for scale-wise object detection, the model has bottom-up and top-down paths to improve to the detection.

Zhao *et al.* [1] introduced a multilevel FP network to construct more accurate FPs for identifying objects in different scales. Yu *et al.* [5] introduced a FP model based on pyramid pooling [34] and feature transformation. Li *et al.* [26] and Yang *et al.* [35] presented encoder–decoder pyramid networks based on sparse coding for cloud detection in an optimal and efficient way. The model extracts multiscale contextual information without the loss of resolution. Sun and Wu [12] proposed a model that used spatial pyramid attentive pooling for syntactic image generation that can be integrated into both generators and discriminators. Pang *et al.* [36] introduced a pyramid style model to produce featurized images in a single-stage detection framework. The multiscale features are then added into the prediction layers of the detector using an attention module.

III. Proposed Method

We intend to develop a road segmentation model in the source domain that has labeled data sets and generalizes the system to the target domain that has an unlabeled data set. In this section, we first introduce the background of GANs and describe the design of the proposed ASPN model that has a domain adaptation structure. Then, we discuss the details of the proposed architecture.

A. Background

GANs generally contain a generator (G) that trace random noise, (n) to create samples and a discriminator, (D) that recognizes the fake samples from the real ones. The basic framework of the conditional GAN can be observed as a game between G and D, to figure out the equilibrium to the min–max problem

$$\min_G \max_D E_{x\ q(x)}\left[\log D(x)\right] + E_{n\ p(n)}\left[\log(1 - D(G(n)))\right] \quad (1)$$

while $n \in R^{dn}$ is a hidden variable from the distributions of $N(0, 1)$ or $U[-1, 1]$. To generate a model based on the input images, DNNs are typically applied in both G and D. In the proposed architecture, the generator is conditioned based on the additional features maps x^s of the synthetic images. While training, the generator $G(x^s, n; \theta_G) = x^s_{\text{conv4}} + \hat{G}(x^s, n; \theta_G)$ converts the synthetic feature maps x^s and noise map n to an adjusted feature map x^{fm}. It is noticeable that the residual presentation $\hat{G}(x^s, n; \theta_G)$ between the fourth convolution layer's feature maps of real and synthetic images, rather than straightforward computation x^{fm} is computed by the generator.

It is expected that x^{fm} keeps the real semantics of the base feature map x^s. Hence, x^f is fed to the discriminator $D(x, \theta_D)$ and to the classifier $T(x, \theta_T)$. Here, the task of discriminator $D(x, \theta_D)$ is to recognize the converted feature maps x^{fm} that are created by the generator, from the real image's feature maps of the target domain x^t, at the last stage. The classifier division $D(x, \theta_T)$ assigns labels to each pixel in the input image, which is called the deconvolution layer [36]. Fig. 1 illustrates the overall structure of the proposed ASPN model. The objective is to optimize the following min–max function

$$\min_{\theta_G \theta_T} \max_{\theta_D} = \ell_d(G, D) + \partial \ell_t(G, T) \quad (2)$$

while ∂ is a weight that operates the arrangement of the losses. ℓ_d denotes the domain loss

$$\ell_d(D, G) = E_{x^t}\left[\log D\left(x^t, \theta_D\right)\right] + E_{x^s, n}\left[\log\left(1 - D\left(G\left(x^s, n; \theta_G\right), \theta_D\right)\right)\right]. \quad (3)$$

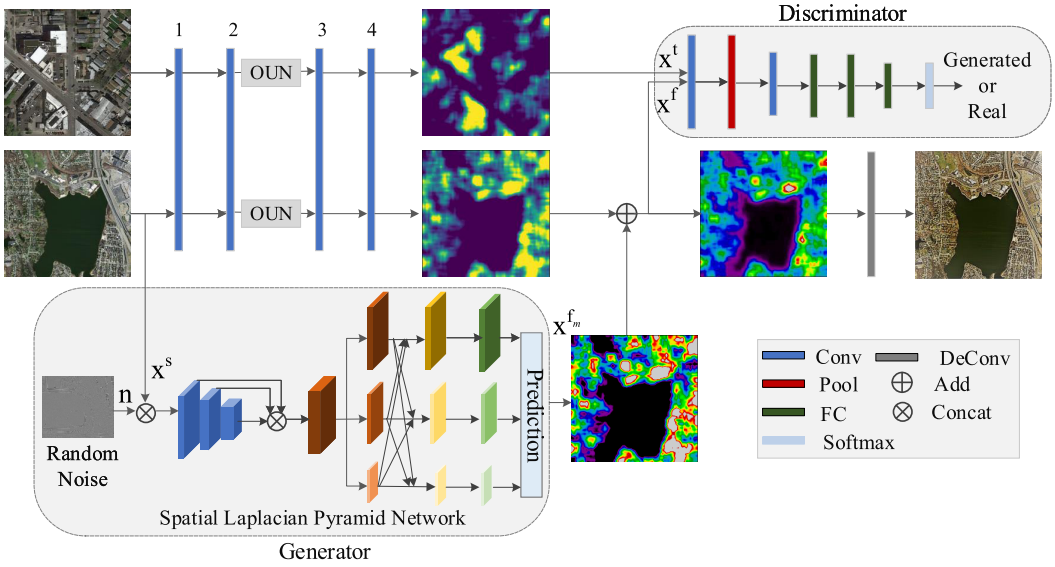

Fig. 1. Overall architecture of the proposed pyramid domain adaption network.

In the deconvolution division, we introduce the function loss ℓ_f as a multinomial loss (cross-entropy loss)

$$\begin{aligned}\ell_f(G,T) &= E_{x^s,y^s,n}\Bigg[-\sum_{i=1}^{|P^s|}\sum_{k=1}^{S^c} 1^{y_i=k}\log\left(T\left(x_i^s,\theta_T\right)\right) \\ &\quad -\sum_{i=1}^{|P^s|}\sum_{k=1}^{S^c} 1^{y_i=k}\log\left(T\left(G\left(x_i^s,n,\theta_G\right),\theta_T\right)\right)\Bigg] \quad (4)\end{aligned}$$

while $\sum_{i=1}^{|P^s|}$ and $\sum_{j=1}^{S^c}$ determine the summary of $|P^s|$ pixels and S^c semantic classes, $1^{y_i=k}$ is a hot encoding of the ith pixel. Multiscale pyramid networks with multiple transformations are used in G. Multilayer perceptron is implemented in discriminator D. The min–max optimization is sought between two levels. Through the first level, the discriminator D and the pixel-wise classifier T are frequently updated, while the conditional generator G and the feature extractor Convolution 1–4 and a single OUN remain intact. Within the next level, T and D are stable while we update G, OUN, and Convolution 1–4. It should be considered that T is trained with both extracted and inferable base feature maps. Training T on the extracted feature maps particularly leads to similar efficiency, whilst demanding several rounds of initializations and multiple learning rates because of the stability issues of the GAN. In fact, if the model does not get trained on the source data, there is the possibility of the shift class assignments (e.g. class 1 change to 2), and the objective function remains optimized. Teng *et al.* [13] proposed to train classifier T equally on the source and adapted images, which eliminate the class shifts and significantly stabilize training. In general, both G and D are cyclewise repeated while G accepts a random prior $z \in R^r$ to optimize the individual parameters, θ_g and θ_d are generated as follows:

$$\begin{aligned}\theta_d &\leftarrow \theta_d + S\nabla\theta_d \frac{1}{m}\sum_{i=1}^{m}\log D(x_i) + \log(1 - D(G(z_i))) \\ \theta_g &\leftarrow \theta_g - S\nabla\theta_g \frac{1}{m}\sum_{i=1}^{m}\log(1 - D(G(z_i))) \quad (5)\end{aligned}$$

in which m is the minibatch size and s is the step size. Indeed, G can be trained to take full advantage of $\log(D(G(z))$ rather than reducing $\log(1 - D(G(z))$ to deliver stronger gradients at the beginning of the training

$$\theta_g \leftarrow \theta_g - S\nabla\theta_g \frac{1}{m}\sum_{i=1}^{m}\log(D(G(z_i))). \quad (6)$$

Therefore, (6) is applied in our model as its performance is more stable. By using the base network for pre-training, GAN has the ability to produce more distributed records. In this work, we propose OUN (the details are presented in the next section) that has encoding/decoding paths. Therefore, the pretrained decoder Dec can choose the appropriate pixels from $G(z)$ for recovering the image $\mathrm{Dec}(G(z))$. For the given input, D is trained to distinguish a synthetic sample $\mathrm{Dec}(G(z))$ from a real one x. The proposed model is trained as follows:

$$\begin{aligned}\theta_d &\leftarrow \theta_d + S\nabla\theta_d \frac{1}{m}\sum_{i=1}^{m}\log D(x_i) + \log(1 - D(x_i)) \\ \theta_{g,\mathrm{dec}} &\leftarrow \theta_{g,\mathrm{dec}} - S\nabla\theta_{g,\mathrm{dec}} \frac{1}{m}\sum_{i=1}^{m}\log(D(x_i)) \\ &\text{in which } x_{z_i} = \mathrm{Dec}(G(z_i)). \quad (7)\end{aligned}$$

The aim of G is to generate samples that D is not able to distinguish them from the real ones. Therefore, G can learn to map different random priors z into the same synthetic output, instead of creating diverse synthetic outputs. This issue is denoted as "mode collapse" that happens when the GAN's

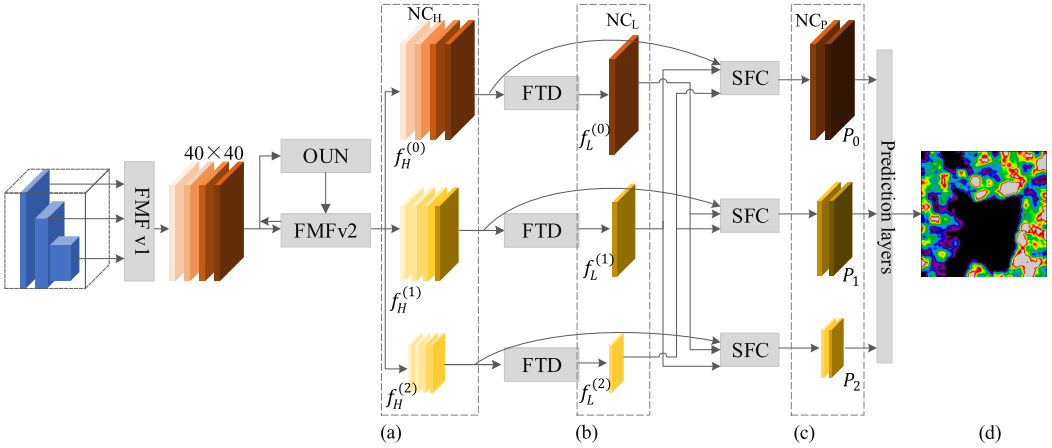

Fig. 2. Architecture of the proposed FP network. (a) High-dimension features. (b) Low-dimension features. (c) FP. (d) Feature map.

optimization approach is solving the min–max problem. Minibatch averaging is used in our model, which is motivated by the philosophy of minibatch discrimination [37]. It provides D with the access to the minibatch of the real samples and the fake samples $(x_1, x_2, \ldots,$ and $G(z_1), G(z_2)$ respectively), whereas classifying samples. Given a sample to discriminate, minibatch computes the gap between the given sample and the other samples in the minibatch. Minibatch shows the average of the minibatch samples to D, hence the objective is modified as follows:

$$\theta_d \leftarrow \theta_d + S\nabla\theta_d \frac{1}{m}\sum_{i=1}^{m} \log D(x_i, \overline{x}_z) + \log\left(1 - D\left(x_{z_i}, \overline{x}_z\right)\right)$$

$$\theta_{g,\text{dec}} \leftarrow \theta_{g,\text{dec}} - S\nabla\theta_{g,\text{dec}} \frac{1}{m}\sum_{i=1}^{m} \log\left(D\left(x_{z_i}, \overline{x}_z\right)\right)$$

in which

$$\overline{x} = \frac{1}{m}\text{Dec}(G(z_i)), \quad \overline{x}_z = \frac{1}{m}\sum_{i=1}^{m} x_{z_i} \tag{8}$$

while m represents the size of the minibatch. Consequently, the average of the minibatch is concatenated on the provided samples and fed to D.

B. Generator Architecture

In particular, the aim of the generator is to produce images similar to real ones for synthetic images to minimize the domain gap. To achieve this goal, we propose a FP network with multiple connections to augment the representations of the synthetic images, as well as several transformations that are helpful for detecting multiscale objects and generating accurate feature maps. The feature maps in the FP that are based on the bottom-up convolution network cannot fulfill the demands of the upper-level feature maps that are achieved by the deep transformation functions; in contrast, the lower-level feature maps acquired by the shallow transformation units also cannot deliver across the second property that limits the detection performance on small objects. Moreover, each generated feature map just represents the output of its corresponding scale, so relative information of the other scales cannot be successfully compounded. One of the options to prevail over these limitations is to applying the transformation functions with an appropriate depth to preserve the sufficient spatial information as well as the high-level semantic information. The architecture of the proposed FP network is shown in Fig. 2. In the proposed model, the backbone and the multistage FP network are used to extract features, followed by the nonmaximum suppression (NMS) operation to generate feature maps from the input images. The proposed FP network contains four divisions. feature map fusion (FMF), optimized u-shape network (OUN), feature transportation division (FTD), and scale-wise feature concatenation (SFC). FMFv1 enhances the semantic information of the basic features by combining the feature maps of the backbone. The OUN largely produces a collection of multiscale features, and then the FMFv2 is applied to fusing the features produced by OUN with the basic features to extract multistage multiscale features. The concatenated feature maps again are fed to the next OUN (there are six groups of OUN + FMFv2). The first OUN is just learned from X_{ini}. The final multistage multiscale output features are computed as follows:

$$\left[X_1^l, X_2^l, \ldots, X_i^l\right] = \begin{cases} T_l(X_{\text{ini}}), & \text{if } l = 1 \\ T_l F\left(X_{\text{ini}}, X_i^{l-1}\right), & \text{if } l = 2, \ldots \end{cases} \tag{9}$$

while X_{ini} represents the initial features that are extracted from the base network, X_i^l signifies the features with the ith scale in the lth OUN, L represents the total number of OUNs, T_l shows the performance of lth OUN, and F represents the proceeding of FFMv1. Furthermore, SFC gathers the multistage FP via a scalewise feature interpolation function and an adjusted attention approach. The details of each division is shown in Fig. 3 and illustrated as follows.

1) Base Network: Similar to [4], ResNet-101 and VGGNet-16 are adopted as the base network. In the proposed FP network, the last FC layers of ResNet-101 and VGGNet-16 are switched with a convolution layer for subsampling their parameters. In the base network, the output of the lth layer is signified as b_l and the outputs of the backbone network are presented as $B_{\text{net}} = \{b_1, b_2, \ldots, b_L\}$, accordingly the

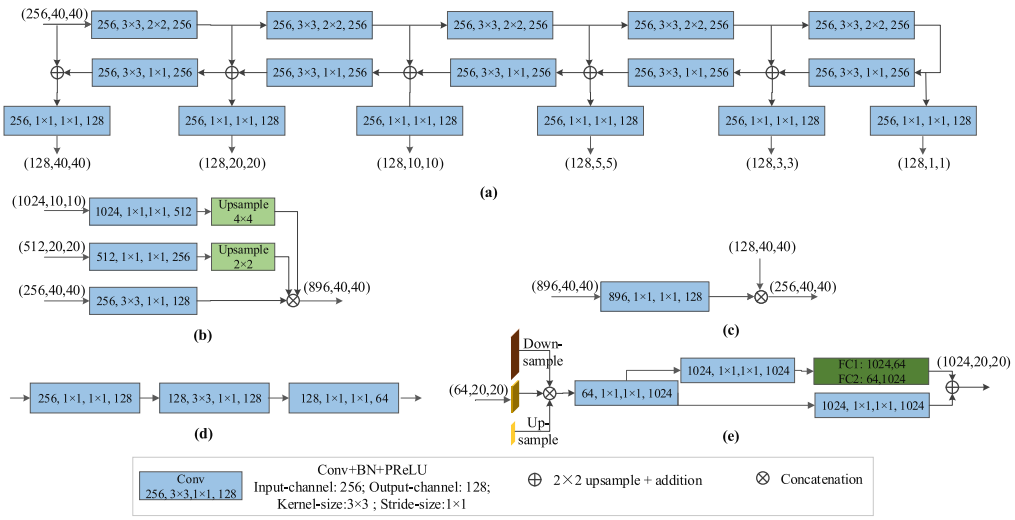

Fig. 3. Detailed structure of several modules. (a) OUN. (b) FMFv1. (c) FMFv2. (d) FTD. (e) SFC. The numbers inside the blocks denote input channels, Conv kernel size, stride size, output channels.

prediction feature map sets can be represented as follows:

$$B_{\text{pred}} = \{b_P, b_{P+1}, \ldots, b_L\} \tag{10}$$

while $P \gg l^3$, b_L are deep feature maps. When $P < l < L$, b_l goes to the shallow feature maps and extracts low-level features. The high-resolution maps with partial semantic information may not well lead to object detection and segmentation. In our observation, reusing deep and shallow semantic information is the key bottleneck in increasing the performance of the model. To enhance the shallower layers' semantic information, we can add the features from the deeper layers. For example,

$$\begin{aligned} \acute{b}_L &= b_L \\ \acute{b}_{L-1} &= w_{L-1}.b_{L-1} + \alpha_{L-1}.b_L \\ \acute{b}_{L-2} &= w_{L-2}.b_{L-2} + \alpha_{L-2}.\acute{b}_{L-1} \\ &= w_{L-2}.b_{L-2} + \alpha_{L-2}w_{L-1}.b_{L-1} + \alpha_{L-2}\alpha_{L-1}.b_L \end{aligned} \tag{11}$$

where w and α are weights. Without considering the generalization loss:

$$\acute{b}_l = \sum_{i=p}^{L} W_l.b_l \tag{12}$$

where W_l denotes the generated weights for the output of the lth layer and the final features are expressed as

$$\acute{B}_{\text{pred}} = \left\{\acute{b}_P, \acute{b}_{P+1}, \ldots, \acute{b}_L\right\}. \tag{13}$$

From (12), it can be observed that the final features ($\acute{b}_l$) are corresponding to the merging ($b_l, b_{l+1}, \ldots, b_L$). One of the methods to enhance the shallow layers's information is the linear combination with the deeper feature hierarchy.

2) FMF: The task of this division is to fuse the features that are extracted from different levels and have a rule in constructing the last multistage pyramid feature map. First, 1×1 convolution layers are used to compact all the channels of the input features and second, to combine these feature maps, concatenation is applied. In particular, FMFv1 receives three different scales' feature maps from the backbone network as the input and it has two different scale upsampling functions to rescale the deep features to the equal scale before concatenating the features. On the other hand, FMFv2 receives two same scale feature maps as the input. One is the base feature and the other one is the largest output feature map of the earlier OUN and we generate the fused feature for the next OUN. The detailed structures of these two divisions are presented in Fig. 3(b) and (c), respectively.

3) OUN: In contrast to other FP networks [1], [4], in the proposed FP network, OUN is used, and the details are presented in Fig. 3(a). The downsampling path has five sequences of 3×3 convolution layers with stride 2. The upsampling path receives the outcomes of different layers as its reference set, however, the other approaches just receive the result of the final layer of each stage in the residual network backbone [36]. Furthermore, to increase the learning performance and preserve the smoothness for the features, 1×1 convolution layers are added after each upscaling and addition process in the upsampling path. In each OUN (six are used), all the outputs in the upsampling path configure the multiscale features at the present stage. Overall, the stacked of OUNs build the multistage multiscale features, from the shallow to the deep level features.

4) FTD: In FTD, for computational efficiency, three convolution layers are implemented with the size of 1×1, 3×3 and again 1×1 to decrease the channel number. Moreover, for input normalization and activation, the batch normalization (BN) and the parametric rectified linear unit (PReLU) are used. 1×1 convolution in the FTD reduces the special feature maps by half channels $\text{NC}_H = D$, while NC is the number of the output channels of the FTD.

5) Feature Pyramid Pooling: In classification and segmentation tasks, pooling layers are widely used,[1] and these layers

[1] http://deepglobe.org/, 2018.

not only spatially decrease the size of the feature maps, but also combine the contextual information of the subregions. He *et al.* [34], introduced a model that use different subregions pooling sizes to generate FP for object detection and segmentation. If the base network generates the $W \times H$ size feature map with D channels, first to each high dimensional FP, pooling is applied, $F_H = \{f_H^{(0)}, f_H^{(1)}, \ldots, f_H^{(N-1)}\}$, while $f_H^{(n)}$, with the feature map spatial size of $(W/2^n) \times (H/2^n)$, represents the nth stage of F_H, and N signifies the level of the pyramids. Hence, the downsampling of the feature maps is used to reduce the spatial size by half. Later, the FTD is used to decrease the channel numbers. Meanwhile, the output of the FTD with low-dimensional FP pooling is represented as $F_L = \{F_L^{(0)}, F_L^{(1)}, \ldots, F_L^{(N-1)}\}$ while the reduced channel number is $C_L = D/(N-1)$.

6) SFC: The rule of this layer is to combine the multiscale features that are stepwise produced by OUNs to generate the multiscale FP maps. The details are presented in Fig. 3(e). First, the SFC concatenates the same scale features together with the channel dimension. As the SFC units reuse the feature maps in F_L, efficiently, we can extract the different scales' context information to derive P_n with NC_P channels. In SFC, the feature maps from F_H are aggregated with feature maps of F_L by the help of skip connection. Hence, the numbers of the output channels reach $2\mathrm{D}/N$, therefore, the aggregated feature maps have 2D channels. The concatenated FPs can be denoted as $X = [X_1, X_2, \ldots, X_i]$, while $X_i = \mathrm{Concat}(X_i^1, X_i^2, \ldots, X_i^L) \in R^{(W_i \times H_i \times \mathrm{NC})}$ states the features of the ith largest size. After applying the aggregation to different scale FPs, the output of this division contains features from several layers depths. However, just applying simple aggregations is not sufficient. Therefore, the channel-wise attention module is used to enforce the features to focus on the most representative channels. Global average pooling [38] is used to create channel-wise statistics $\mathrm{GP} \in R^{\mathrm{NC}}$ at the combining section. To precisely capture the channel-wise dependencies, two fully connected layers are used as follows:

$$A = F_{\mathrm{ex}}(\mathrm{GP}, W) = \rho\left(W_2^{\sigma}\left(W_1^{\mathrm{GP}}\right)\right) \tag{14}$$

while ρ denotes the PReLU function, σ represents the softmax function, $W_1 \in R^{(\mathrm{NC})/r \times \mathrm{NC}}$, $W_2 \in R^{\mathrm{NC} \times (\mathrm{NC})/r}$, and r denotes the decreasing ratio (in this work $r = 8$). The last output is computed by reweighting the input X with activation A, while $\tilde{X}_i = [\tilde{X}_i^1, \tilde{X}_i^2, \ldots, \tilde{X}_i^c]$ and the rescaling process either helps the features to become more accurate or weakened

$$\tilde{X}_i^{\mathrm{NC}} = F_{\mathrm{scale}}\left(X_i^{\mathrm{NC}}, A_{\mathrm{NC}}\right) = A_{\mathrm{NC}}.X_i^{\mathrm{NC}}. \tag{15}$$

Prior to the whole network training, we pretrain the backbone on the data sets. The input size to the proposed FP network is 320×320 and the network contains 6 OUNs, where each OUN has 5 convolution layers in the down-sampling path and 5 in the up-sampling path, and therefore the output features have six different scales. To decrease the number of the parameters, we only assign 256 channels to each scale of their OUN features; hence, the network can simply train on GPUs. Two convolution layers are added to each of the six pyramidal features to achieve location regression and perform classification. At every pixel of the pyramidal features, six anchors with three ratios are set and a probability score (threshold) of 0.05 to cut out anchors that have low scores. Reducing the threshold to 0.01 can improve the performance; however, it also increases the processing time.

C. Discriminator Architecture

As Fig. 1 shows the discriminator is trained to distinguish the created feature for the source domain images and the real ones from the target domain images. It receives the vectorized feature maps as input and pass them to the three FC layers followed by a classification layer with the softmax function. The output dimensions of the three FC layers are 4096, 4096, and 1024. By differentiating between the fake and real image representations, an adversarial loss is proposed to help the generator to create the representation for the synthetic images that are quite similar to the genuine ones.

D. Real Images Testing

In the testing part, we used our previous work reported in [39]. A real image goes over the feature extractor, Convolution and OUN layers followed by the pixelwise classifier for road segmentation [39]. The generator and discriminator will not participate and the proposed architecture should almost have similar interpretation complexity. The model achieves 4.48 fps with one NVIDIA GTX GeForce 1080 Ti GPU. To discuss the computation cost of ASPN, first, we consider a two scale pyramid network [36]. Here, $I(j \times j)$ denotes an image, $l = d(I)$ represents the coarsened image, and $h = I - u(d(I))$ calculates the high pass. To streamline the computations, a different u operator is used to produce the images. We have taken $d(I)$ as the mean of each disjoint block of 2×2 pixels, and u as the factor that eliminates the mean from all the 2×2 blocks. Subsequently, u has the rank of $3d^2/4$, here, we assign h as an orthonormal basis to the range of u, therefore, the linear mapping between I and (l, h) is unitary. For creating a probability density p on R^{d^2}, we follow:

$$p(I) = q_0(l,h)q_1(l) = q_0(d(I), h(I))q_1(d(I)) \tag{16}$$

if $q_i \geq 0$, $\int_a^b q_1(l)dl = 1$, and for every constant l, $\int_a^b q_0(l,h)dh = 1$. To verify whether or not p has a unit integral

$$\begin{aligned} \int pdI &= \int q_0(d(I), h(I))\ q_1(d(I))\ dI \\ &= \int\int q_0(l,h)\ q_1(l)\ dl\ dh = 1. \end{aligned} \tag{17}$$

A set of training samples $(l_1, \ldots, l_{N0})$ are taken for q_1, and the density function is accordingly constructed as follows:

$$q_1(l) \sim \sum_{i=1}^{N_1} e^{\|l-l_i\|^2/\sigma_1}. \tag{18}$$

To define $q_0(I) = q_0(l,h) \sim \sum_{i=1}^{N_0} e^{\|l-l_i\|^2/\sigma_1}$, we set $l = d(I)$. A similar method is used for each of the finer scales while the pyramid network has more levels. It should be considered that generally the low pass filter at each scale is

used, and additionally we measure the true high pass versus the generated high pass samples of the model. Therefore, using a pyramid that has M levels, the last log likelihood is computed as follows:

$$\log(q_M(l_M)) + \sum_{M=0}^{M-1} \log(q_M(l_M, h_M)). \tag{19}$$

The training pseudocode of ASPN is shown as follows.

Algorithm 1 ASPN Training Pseudocode

for number of training iterations **do**
 for k steps **do**
 • Draw a minibatch of samples $\{x_1, \ldots, x_m\}$ from generated data distribution.
 • Draw a minibatch of noise samples $\{G_{z_1}, \ldots, G_{z_m}\}$ from noisy prior (n).
 • update the discriminator:
 $s\nabla\theta_d \frac{1}{m} \sum_{i=1}^{m} \log D(x_i, \bar{x}) + \log(1 - D(x_{z_i}, \bar{x}_z))$
 end for
 • Dec is the pre-trained decoder which chooses the appropriate pixels from $G(z)$ for image recovery $(x_{z_i} = Dec(G(z_i)))$.
 • update the generator:
 $s\nabla\theta_{g,dec} \frac{1}{m} \sum_{i=1}^{m} \log D(x_{z_i}, \bar{x}_z))$
 • Update the feature extractor, source domain classifier and target domain classifier:
 $l_f(G, T) = E_{x^s, y^s, n}[-\sum_{i=1}^{|p^s|} \sum_{k=1}^{s^c} 1^{y_i=k}$
 $\log(T(x_i^s, \theta_T)) - \sum_{i=1}^{|P^s|} \sum_{k=1}^{s^c} 1^{y_i=k}$
 $\log(T(G(x_i^s, n, \theta_g), \theta_T))]$
end for
We use $\theta_g \leftarrow \theta_g - s\nabla\theta_g \frac{1}{m} \sum_{i=1}^{m} \log(D(G(z_i)))$ to deliver stronger gradients at the beginning of the training.

IV. Experimental Details and Analysis

In the following section, we explain the details of the used data sets, experimental results, and compare the performance of the ASPN model with that of the other models.

A. Data Sets and Experimental Setup

To accurately evaluate the performance of the proposed model, three road detection data sets are used in our experiments. These data sets have different distributions and densities of objects captured from different distances. To improve the diversity of the training samples, we have the following alterations for data augmentation.

1) Random vertically and horizontally flipping.
2) Random conversion by $[-8, 8]$ pixels.
3) Random scaling in the range $[1, 1.5]$.

1) DeepGlobe Road Extraction Data Set[2]*:* It is a 2-tiles data set from India, Thailand, and Indonesia. The road extraction data set has complex road environments. The data set holds 6226 training images and 2344 testing and validation images. Each image has a size of 1024×1024, and the ground resolution of the image pixels is 0.5 m/pixel. We used all the training samples to pre-train the backbone and 1169 samples for training the network and ignored repetitive images.

2) Massachusetts Road Data Set[3]*:* The data set consists of 1171 images of the state of Massachusetts. The size of each image is 1500×1500 pixels with a spatial resolution of 1 m per pixel, composed of red, green and blue channels. This data set was collected from aerial images. The ground truth of the images consists of two classes, roads and nonroads.

3) EPFL Road Segmentation Data Set[4]*:* The data set consists of 150 aerial images. The size of each image is 400×400 pixels with a spatial resolution of 1 m per pixel.

The Massachusetts validation and test sets are used for the proposed model validation and testing. To train the model, random mini-batches are selected from the images (and their labels) of the source domain data set and the real images (without labels) from the target domain data set. In addition, 20 images from the testing set of the EPFL road data set are used to test ASPN. The proposed model is evaluated based on Fréchet inception distance (FID) [40]. The samples from X and Y are inserted into a feature space by using a particular Inception network. These feature distributions are formed as multidimensional Gaussians parameterized by their individual covariance and mean. Then the FID is measured by

$$\text{FID} = \|\mu_x - \mu_y\|^2 + \text{Tr}\left(\sum_x + \sum_y - 2\left(\sum_x \sum_y\right)^{\frac{1}{2}}\right) \tag{20}$$

while $(\mu_x, \sum_x)$ and $(\mu_y, \sum_y)$ represent the mean and covariance distribution of the real and generated images, respectively. We adopted the FID score to evaluate the performance of the generative model on the unlabeled data. Additionally, the code released with the Cityscapes data set [11] is also used to evaluate the results. It computes the intersection-over-union $= (\text{TP})/(\text{TP} + \text{FP} + \text{FN})$, while TP, FP, and FN are the numbers of true positive, false positive, and false negative pixels, respectively, that are calculated over the test set. Additionally, we used a sematic instant-level Intersection-over-union metric (iIoU), which represents how well each instance in the scene is signified in the labeling.

To fairly evaluate the performance of ASPN, several evaluation metrics have been chosen, such as the variation of probabilistic rand index (PRI) [41], information metric (VOI) [42], global consistency error (GCE) [43], and boundary displacement error (BDE) [44]. Based on these parameters, if a segmentation result is good, whenever the comparison with ground truth yields a high value for PRI and small values for the other three metrics. PRI can be defined in a simple form. Let S_{ground} and S_{test} be two clusters of the same image, and n_{ij} is the sum of points in the ith cluster of S_{ground} and the jth cluster of S_{test}. N signifies the total number of the pixels of the image. The value of PRI, which measures the similarity of two clusters, ranges between 0 and 1 and can be formulated

[2]http://deepglobe.org/, 2018.
[3]https://www.cs.toronto.edu/ vmnih/data/, 2013.
[4]https://www.kaggle.com/c/epfml17-segmentation/leaderboard, 2017.

as follows:

$$\mathrm{PRI}\left(S_{\text{ground}}, S_{\text{test}}\right) = \left\{ \binom{n}{2} - 1/2 \left\{ \sum_i \left(\sum_j n_{ij}^2 \right) + \sum_j \left(\sum_i n_{ij}^2 \right) - \sum\sum n_{ij}^2 \right\} \right\} \Big/ \binom{n}{2}. \quad (21)$$

VOI, computes the loss/gain between the images. H and I, respectively, denote the entropies and the mutual information between the two clusters, and defined as

$$\mathrm{VOI}\left(S_{\text{ground}}, S_{\text{test}}\right) = H_{\text{ground}} + H_{\text{test}} - 2I\left(S_{\text{ground}}, S_{\text{test}}\right). \quad (22)$$

GCE, defines the consistency of segmentation. $R(S_{\text{ground}}, p_i) \Delta R(S_{\text{test}}, p_i)$ represents the symmetric difference between $R(S_{\text{ground}}, p_i)$ and $R(S_{\text{test}}, p_i)$. The nonsymmetric local consistency error is calculated as

$$E\left(S_{\text{ground}}, S_{\text{test}}, P_i\right) = \frac{R\left(S_{\text{ground}}, p_i\right) \Delta R(S_{\text{test}}, p_i)}{R\left(S_{\text{ground}}, p_i\right)}. \quad (23)$$

GCE is computed by symmetrization and averaging

$$\mathrm{GCE}\left(S_{\text{ground}}, S_{\text{test}}\right) = \frac{1}{N} \min \left\{ \sum_i E\left(S_{\text{ground}}, S_{\text{test}}, P_i\right), \sum_i E\left(S_{\text{test}}, S_{\text{ground}}, P_i\right) \right\}. \quad (24)$$

BDE measures the average displacement error of boundary pixels on the segmentation outputs. Indeed, it states the error of one boundary pixel as the distance between the pixel and its nearby boundary pixel in the image

$$d(p_i, B_2) = \|p_i - p\|_{p \in B_2}^{\min}. \quad (25)$$

The distance of a boundary point to the boundary set B_2 represented by $p_i \in B_1$ and N_1, N_2 denotes the total number of the points in the boundary sets B_1 and B_2, accordingly BDE is defined as

$$\mathrm{BDE}(B_1, B_2) = \frac{\sum_i^{N_1} d(p_i, B_2)/N_1 + \sum_i^{N_2} d(p_i, B_1)/N_2}{2}. \quad (26)$$

B. Implementation

In our conditional GAN experiment, we use ResNet-101 [46] as our backbone network. We initialize it with VGGNet-16 [47], and the Adam optimizer [48] is used for training. The minibatches contain five images from the source domain and 5 from the target one. ASPN is implemented by using Keras 2.1.2, the deep learning open-source library and TensorFlow 1.3.0 GPU as the backend deep learning engine. Python 3.6 is used for all the implementations. All the implementations of the network are conducted on a workstation equipped with an Intel i7-6850K CPU, a 64 GB Ram and four NVIDIA GTX Geforce 1080 Ti GPU and the operating system is Ubuntu 16.04. All the images are resized to 320×320, thus, the feature map of the first convolution layer output is $[64, 320, 320]$. Therefore, n is a 320×320 matrix, which is sampled based on a regular distribution $n_{ij} \sim u(-1, 1)$. The first convolution layer feature map is concatenated to n as an additional channel, and fed to the proposed pyramid network. To train the proposed pyramid network with the ResNet-101 backbone, the whole training time costs 5 days and with the VGG-16 backbone, the entire training time costs 2.5 days. In Table I, we compare the performance of ASPN with that of the other approaches; we also evaluate the performance of our model with and without the proposed pyramid network. In Table I, we discuss the following aspects: The backbone network types, the initial input size to the network, the model strategy and the test results of the models. We also report the performance of the proposed pyramid network with different settings on the Massachusetts road data set. It is remarkable that the proposed pyramid network with VGG-16 backbone has the best performance, which has outperformed the other state-of-the-art models even with more powerful backbones. IoU of Huang *et al.* [9] is 25.6, IoU of Zhu *et al.* [24] with FCN is 41.7, IoU of Shrivastava *et al.* [45] with ResNet-101 is 44.5 and IoU of Hong *et al.* [11] with VGG-19 is 45.5. The performance of the proposed ASPN model is evaluated based on different network settings (four, six, and eight OUNs while having ResNet-101 or VGG-16 as the backbone network) for two types of input size (512×512 and 320×320). The IoU of ASPN that is assembled with FCN on 512 and 320 input image size is 44.8 and 44.6 respectively. As the results show, the ASPN with six OUNs, and 256 channels has the best performance among the other implementations. Meanwhile, the performance of the ASPN with eight OUNs, is almost equal to that of six OUNs but the computation costs are higher. Additionally, in ASPN, both VGG-16 and ResNet-101 have better performance on the bigger size's input images and the best performance is achieved from the ASPN with VGG-16 on the image size of 512×512.

C. Analysis of Receptive Field in Discriminator

In Fig. 4, we showed some synthesis samples produced by the ASPN, plus the visualization of attention maps after fusing multiscale feature maps. In fact, Fig. 4 shows the effect of fusing multilevel and multiscale features in synthesis image generation. In the discriminator, similar to [23], a 70×70 receptive field is used to examine each structure at different scales. The discriminator uses a random Markov theory for classification. In this approach, even if a single $N \times N$ patch in an image is treated as a fake, then the classifier counts the image as a fake one. The added convolution layers, before fully connected layers in the discriminator, significantly increase the size of the receptive field without increasing the model depth and adding new parameters.

D. Ablation Study

In this section, we evaluate the effects of different modules in the performance of the proposed ASPN model. The results of ASPN and the other state-of-the-art methods are reported in Tables I and II. It has been observed that the domain adaptation models perform better than the other approaches. ASPN

TABLE I

PERFORMANCE COMPARISONS IN TERMS OF MEAN IOU AND IIOU PERCENTAGE, FID, NUMBER OF PARAMETERS (PARAM) AND COMPUTATION SPEED (FLOPS) ON MASSACHUSETTS ROAD TEST DATA SET

Methods	Backbone	Input size	Strategy	IoU	iIoU	FID	Param	FLOPs
Xu and Zhao [10]	FCN	512×512	Residual learning	52.34	40.97	31.2	13.37M	84.28B
Huang et al. [9]	—	800×800	Unified fusion	55.62	43.71	28.6	13.51M	84.75B
Zhu et al. [24]	FCN	256×256	discriminator update base	71.77	60.16	26.63	15.32M	86.98B
Shrivastav et al. [45]	ResNet	55×35	discriminator update base	73.53	62.23	20.42	14.82M	86.44B
Hoffman et al. [23]	FCN	600×600	Cycle-Consistent adaptation	75.15	63.03	19.83	15.49M	87.67B
Sun and Wu [12]	FCN	256×256	Pyramid attentive pooling	75.25	63.13	19.76	14.62M	86.41B
Hong et al. [11]	VGG-19	480×960	Domain adaptation	75.48	63.28	19.52	14.78M	86.73B
Liu et al. [7]	FCN	512×512	Deep U-shape	78.67	63.59	18.53	14.97M	86.85B
Li et al. [2]	—	800×600	Two-Stream Pyramid	78.72	63.65	18.52	14.96M	86.87B
ASPN (Ours)	FCN	512×512	Domain adaptation	74.81	62.31	20.27	14.32M	86.23B
ASPN (Ours)	FCN	320×320	Domain adaptation	74.68	60.95	20.36	14.06M	85.89B
ASPN + 4 OUNs	ResNet	320×320	Multiscale Domain adaptation	77.79	64.56	18.79	14.43M	86.52B
ASPN + 6 OUNs	ResNet	320×320	Multiscale Domain adaptation	77.96	63.81	18.75	14.54M	86.68B
ASPN + 8 OUNs	ResNet	320×320	Multiscale Domain adaptation	77.92	63.80	18.78	14.62M	86.71B
ASPN + 4 OUNs	VGG-16	320×320	Multiscale Domain adaptation	77.85	62.73	18.69	14.34M	86.26B
ASPN + 6 OUNs	VGG-16	320×320	Multiscale Domain adaptation	77.99	62.94	18.66	14.51M	86.47B
ASPN + 8 OUNs	VGG-16	320×320	Multiscale Domain adaptation	77.97	62.93	18.68	14.67M	86.53B
ASPN + 6 OUNs	ResNet	512×512	Multiscale Domain adaptation	78.34	63.37	18.60	15.03M	87.22B
ASPN + 8 OUNs	ResNet	512×512	Multiscale Domain adaptation	78.31	63.34	18.65	15.11M	87.3B
ASPN + 6 OUNs	VGG-16	512×512	Multiscale Domain adaptation	78.86	63.74	18.43	14.89M	86.78B
ASPN + 8 OUNs	VGG-16	512×512	Multiscale Domain adaptation	78.85	63.72	18.51	14.96M	86.84B

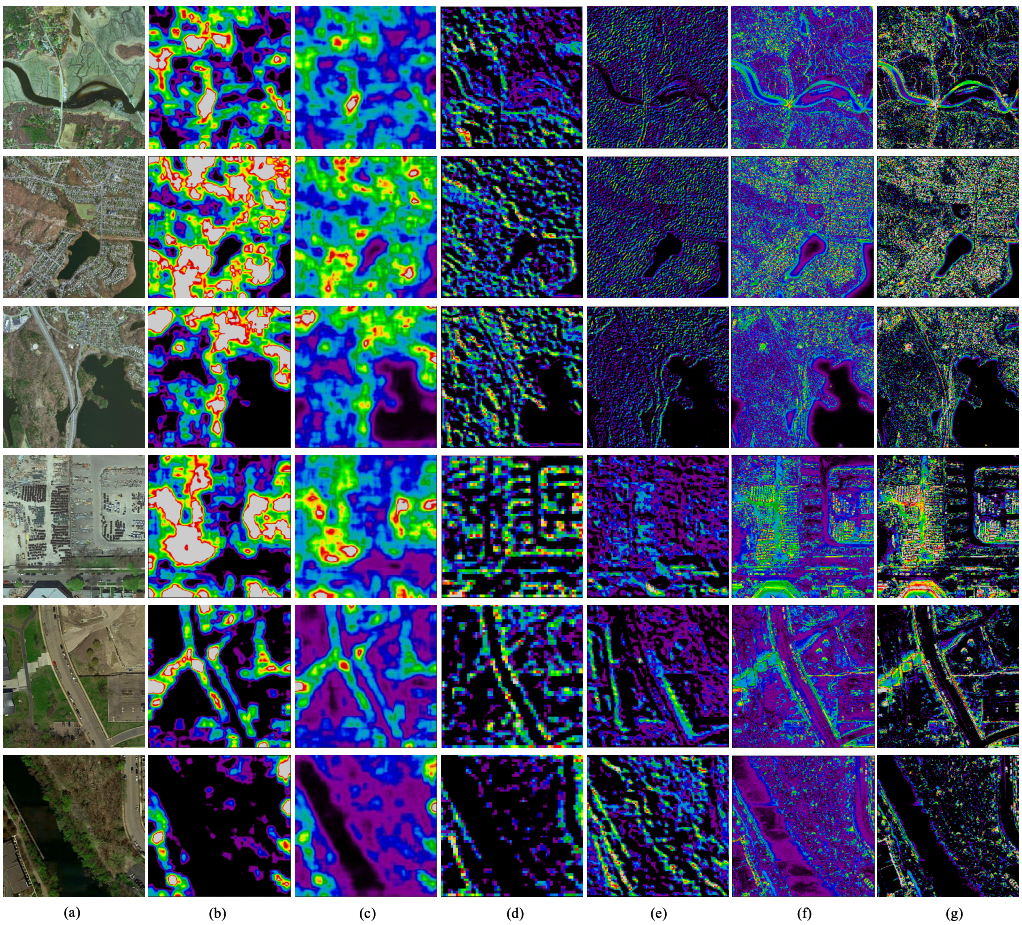

Fig. 4. Samples from synthesized images and visualization of attention map in different modules. (a) Synthesized image. (b) Attention map for C_4 + FP. (c) Attention map for C_1 + FP. (d) Attention map for C_4. (e) Attention map for C_3. (f) Attention map for C_2. (g) Attention map for C_1.

on all three data sets has higher IoU and iIoU as compared to others. The IoU of ASPN, minimum 3.8% is higher than the other baselines and clearly justifies the proficiency of the proposed method. In comparison with the other approaches [11], [12], [23], [45], ASPN outperforms them by a gap of 3.5%∼5.3% in IoU. From Table II and Fig. 5, we note that

TABLE II
EVALUATION OF CONDITIONAL GENERATOR

Datasets		Hoffman et al. [23]	Hong et al. [11]	Li et al. [26]	ASPN Wo G	ASPN W G
EPFL road dataset	IoU	62.3	68.72	71.22	48.53	81.68
	iIoU	50.9	54.26	56.37	39.67	67.43
Massachusetts road dataset	IoU	58.63	76.18	77.15	44.25	78.86
	iIoU	45.37	62.48	62.86	36.78	63.74
DeepGlobe road dataset	IoU	55.34	65.15	66.74	46.24	69.58
	iIoU	43.95	52.63	54.06	37.89	55.61

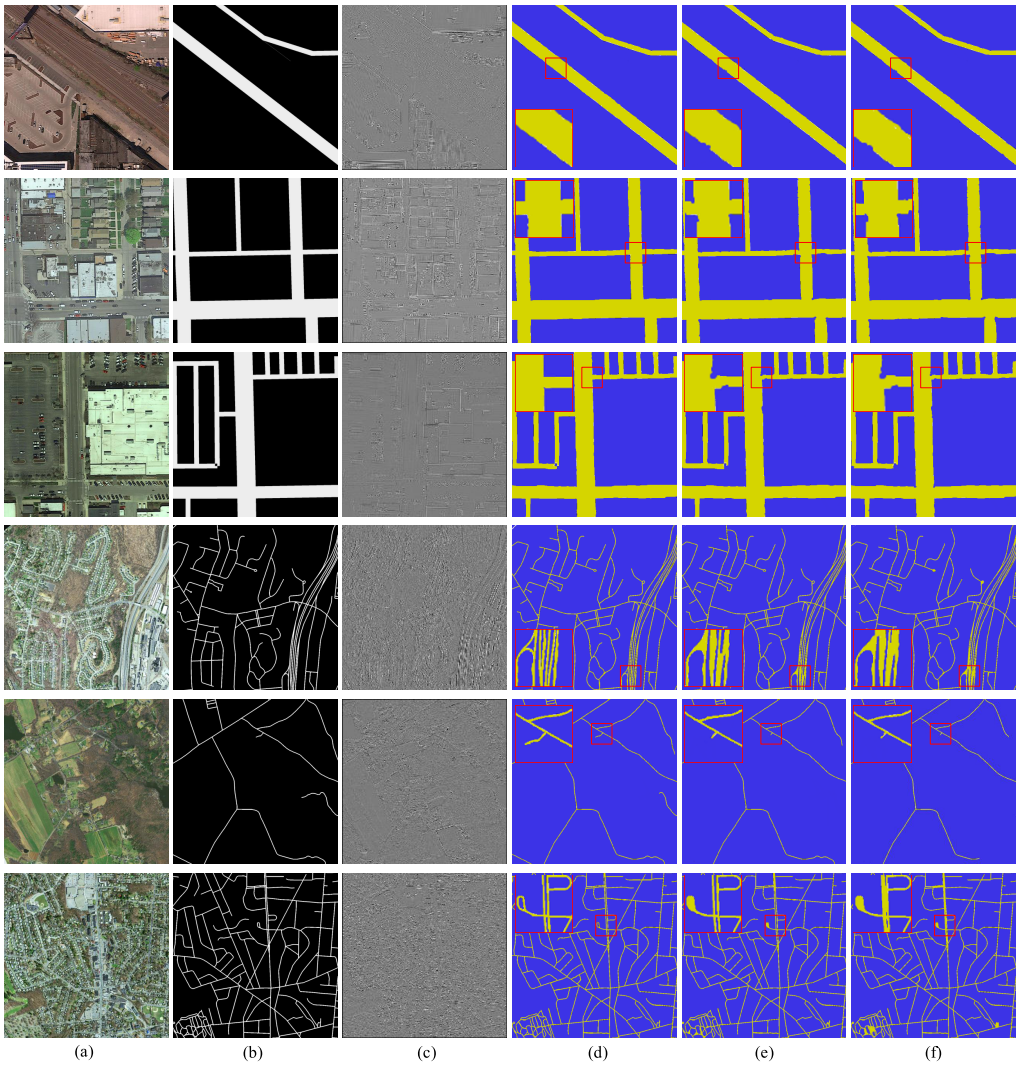

Fig. 5. Qualitative results. (a) Input images. (b) Ground truth annotations. (c) Boundary detection by ASPN (ours). (d) Testing results of ASPN (ours). (e) Testing results of Hong *et al.* [11]. (f) Testing results of Sun and Wu [12]. Red rectangle regions are close-ups for better visualization.

the adaptation from EFPL and Massachusetts are higher than that of DeepGlobe, which is mainly due to a higher number of training images and the structure of the images. Moreover, we show the trend between the segmentation results and the amount of the synthetic data in Fig. 6.

Fig. 6(a) shows the IoUs of the proposed model on all three data sets. It demonstrates that: 1) adopting more synthetic images can significantly improve the performance. For example, 30% increase in the data, can improve the IoU by 9.5%. The best performance of our model is made on the EFPL and Massachusetts data sets respectively as the label maps in these data sets are generated by rasterizing road centerlines, and the average line thickness is about 15–20 pixels with no smoothing. On the other hand, the DeepGlobe data set is an edge-based data set which was annotated based on the width of the road on the images. In this data set, the average width of the road label is about 10 pixels. Therefore, due to slim roads on the images in the DeepGlobe data set, our model does not

TABLE III

ABLATION STUDY OF ASPN IN TERMS OF MEAN IOU PERCENTAGE, MEAN IIOU, AND COMPUTATION SPEED (FLOPS) ON EPFL DATA SET. THE √ SHOWS THE PROPOSED NETWORK DIVISIONS THAT PARTICIPATED IN EACH PART OF EVALUATION

Module	Channels	Base						
+1 OUN	128		√					
	256							
+3 OUN	128			√				
	256							
+6 OUN	128				√		√	√
	256							
+8 OUN	128					√		
	256							
+SFC							√	√
VGGNet-16 to ResNet-101								√
Mean IoU (%)		59.9	71.81	74.52	76.33	76.32	78.26	**81.52**
			72.57	75.28	76.95	76.88	79.51	**81.68**
Mean IoU (%)		48.4	60.63	62.76	65.29	65.28	66.47	**67.29**
			61.07	63.13	66.48	66.49	67.08	**67.43**
FLOPs (B)		30.67	52.24	61.86	68.27	70.95	72.71	80.78
			58.79	70.27	74.83	82.32	83.2	84.42

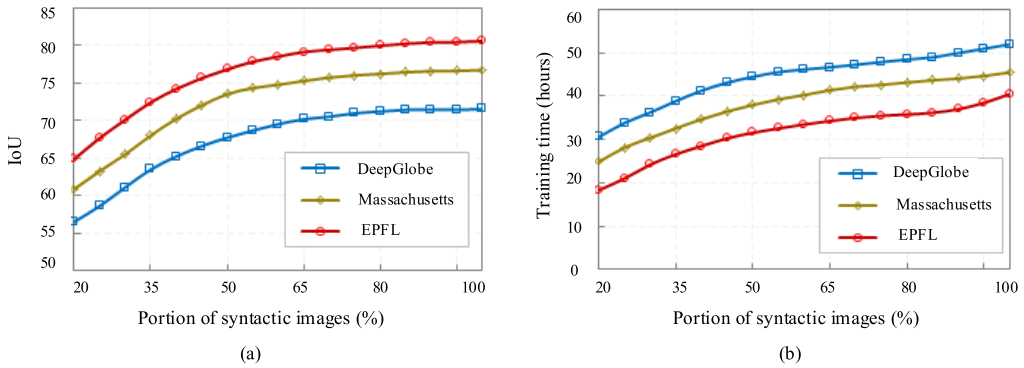

Fig. 6. (a) Change of portions of synthetic images versus IoU. (b) Change of portions of synthetic images versus training time.

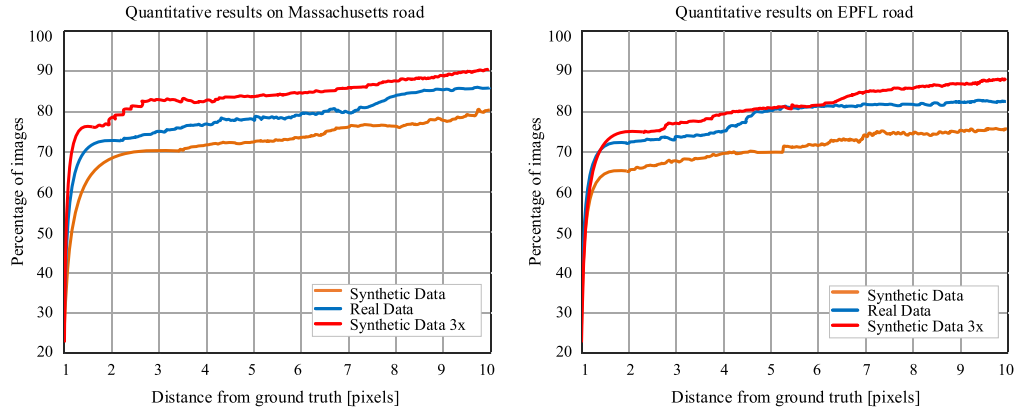

Fig. 7. Quantitative results for road segmentation on the test set of real images from Massachusetts and EPFL data set. The plots represent cumulative curves as a function of distance from ground truth keypoint locations, for several numbers of training samples of synthetic images (3× denotes 100% of the data set).

have high performance on this data set; and 2) in addition to the quantity of the synthetic data, the variety of images is also quite important for training a good model. Fig. 6(b) shows the training time of the ASPN with different amounts of the synthetic data.

1) Influence of the Conditional Generator: To analyze the contribution of the proposed conditional generator toward quality improvement, we have carried out an evaluation for the models that were proposed by Xu and Zhao [10], Hoffman *et al.* [23] and two variations of the proposed

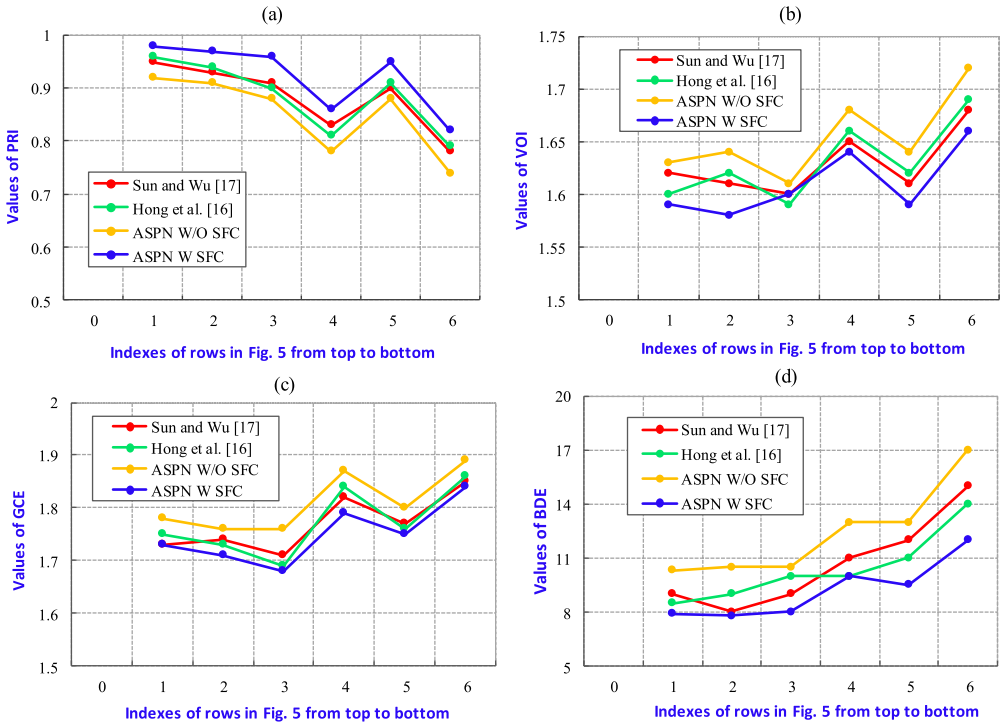

Fig. 8. Quantitative comparisons of the above mentioned two approaches and the proposed ASPN with and without the SFC module on synthetic images with the indexes of rows (from top to bottom in Fig. 5) as the horizontal axis. (a) PRI. (b) VOI. (c) GCE. (d) BDE.

network (trained with and without the conditional generator). In this evaluation, the conditional generator is removed and we only keep the discriminator, the feature extractor, and the pixel-wise classifier. The results are listed in Table II. The conditional generator performs better than other generators over all the three data sets. The results prove the capability of the proposed model in extracting multiscale features from the shallow and deep layers, resulting in high segmentation performance. By removing the conditional generator, both the IoU and iIoU drop around 25%.

2) Influence of Different Modules in Learning: As we have already discussed, ASPN is a collection of various subcomponents and here we analyze the effect of each module.

a) OUN: To validate the effect of this module, three sets of experiments are conducted. First, the base network is extended with a sequence of deconvolution layers. As a result, the IoU has improved from 59.9% to 62.5%. Second, we used the combination of 3 and 6 OUNs in the generator and the performance has been improved to 74.52% and 76.33% on 128 channels, respectively. Finally, we used the stack of 8 OUNs and the best performance reached to 76.32% on 128 channels. To figure out the best configuration for the proposed model regarding the number of OUNs and internal channels, we have conducted several trials. In these implementations, we use the EPFL data set, the backbone network is VGG-16 and the input image size is 320×320. The details are listed in Table III. From the experiments, it has been observed that increasing the number of OUNs and channels can improve the performance of the proposed model. However, it is worth mentioning that, increasing the number of OUNs make it more efficient than increasing the number of internal channels. Even, by adding more number of OUNs, the number of the parameters almost remains constant.

b) SFC: As shown in the 8th column of Table III, by using SFC modules, all the multiscale features are properly concatenated, which greatly helps to improve the performance.

c) Backbone network: From the above tasks, it has been observed, replacing VGGNet-16 with ResNet-101 can slightly improve the performance of the proposed model. As shown in Table III, the IoU goes from 81.52% to 81.68% by adopting the ResNet-101 as the base network. However, it increases the computation cost.

d) Quantitative results: We used our previous work [39] to extract the roads from the synthetic samples. Fig. 7 shows the performance of road segmentation model [39] that is trained on the synthetic data and the real data. As the results show, increasing the number of the training data can improve the results. In this evaluation, the model is trained on the real data, half of the synthesis data and finally, all of the synthesis data. By using the entire synthesis data, there is around 11.8% performance improvement. Fig. 8 shows the quantitative comparisons of the PRI [41], VoI [42], GCE [43], and BDE [44] for the four comparison approaches with the indexes of rows [Fig. 5 (from top to bottom)] as the horizontal axis. From the four charts of Fig. 8, it can be observed the values of the four performance measures for our proposed ASPN model with the SFC module are better than the other two approaches and our model without the SFC module, are slightly worse than the other approaches with multiscale feature network.

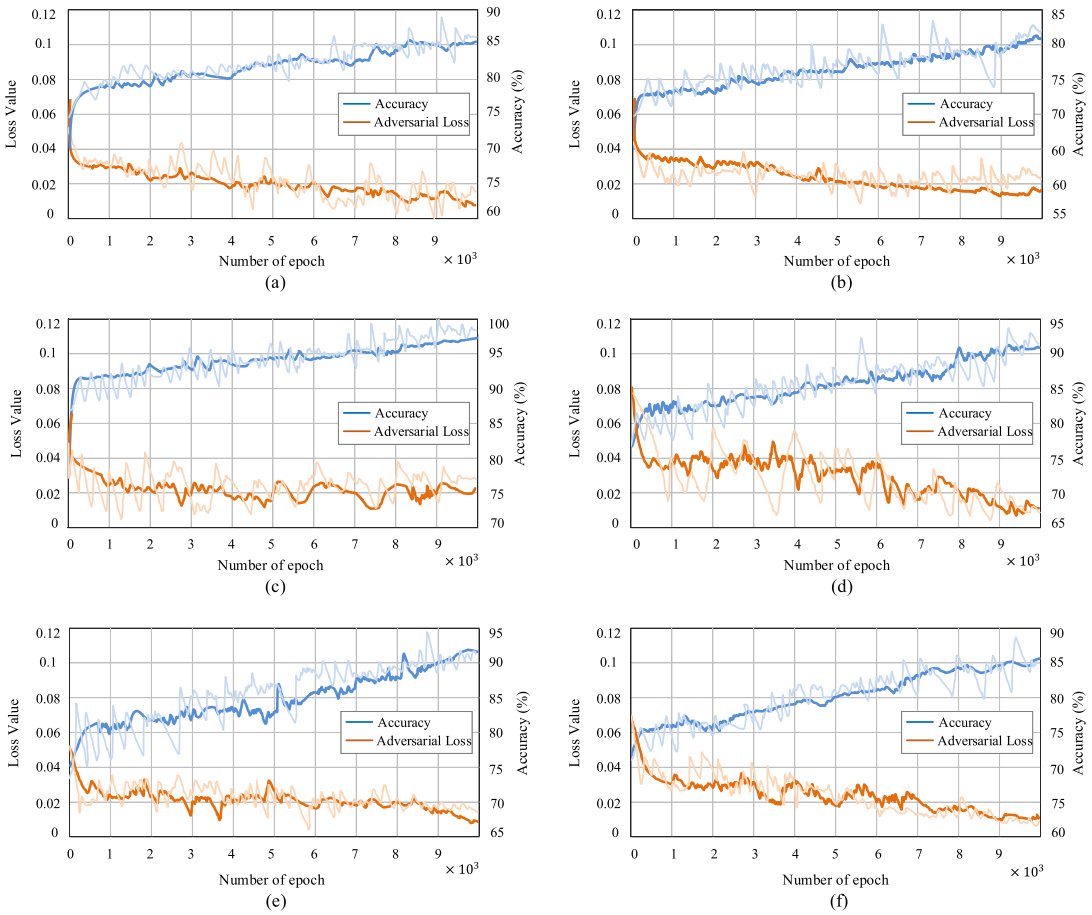

Fig. 9. Relationship between accuracy and adversarial loss while applying cross-domain training on EPFL ↔ Massachusetts, EFPL ↔ DeepGlobe, and Massachusetts ↔ DeepGlobe data sets. (a) EPFL ↔ Massachusetts. (b) EPFL ↔ DeepGlobe. (c) Massachusetts ↔ EPFL. (d) Massachusetts ↔ DeepGlobe. (e) DeepGlobe ↔ EPFL. (f) DeepGlobe ↔ Massachusetts.

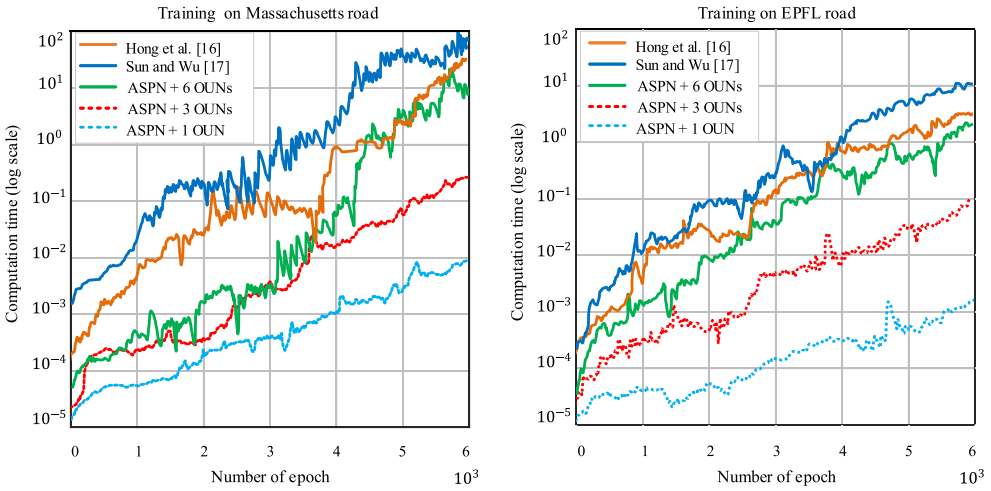

Fig. 10. Computation time versus a number of iterations while training on Massachusetts and EPFL data set. The light blue, red and green refer to propose model with 1, 3 and 6 OUNs respectively. The orange shows the performance of Sun and Wu [12] and dark blue represents the performance of Hong *et al.* [11].

e) Domain adaptation: Fig. 9 demonstrates the relation between the accuracy and adversarial loss during the cross-domain training of the six data sets. EFPL ↔ Massachusetts, EPFL ↔ DeepGlobe, and Massachusetts ↔ DeepGlobe. The results indicate that, applying cross-domain training helps to minimize the adversarial loss and also increases the adaptation accuracy. In this evaluation, the best results are achieved from Massachusetts → EPFL, because, the number of the training

TABLE IV

PERFORMANCE COMPARISON OF SEGMENTATION METHODS ON THE MASSACHUSETTS DATA SET. THE BEST TWO RESULTS ARE HIGHLIGHTED IN GREEN (BEST) AND RED (SECOND BEST)

Methods	PRI	VOI	GCE	BDE
Henry et al. [3]	0.8227	1.7048	0.1938	17.37
Li et al. [6]	0.8288	1.6911	0.1851	12.69
Yu et al. [5]	0.8329	1.6885	0.1837	12.07
Zhang et al. [32]	0.8336	1.6862	0.1822	11.68
Sun et al. [12]	0.8352	1.6853	0.1810	11.34
Hong et al. [11]	0.8356	1.6851	0.1804	11.13
Li et al. [26]	0.8358	1.6845	0.1798	11.06
Liu et al. [7]	0.8359	1.6839	0.1795	10.58
Wei et al. [22]	0.8360	1.6836	0.1791	10.56
Li et al. [2]	0.8362	1.6844	0.1789	10.62
ASPN	0.8367	1.6833	0.1784	10.58

TABLE V

PERFORMANCE COMPARISON OF SEGMENTATION METHODS ON THE EPFL DATA SET. THE BEST TWO RESULTS ARE HIGHLIGHTED IN GREEN (BEST) AND RED (SECOND BEST)

Methods	PRI	VOI	GCE	BDE
Henry et al. [3]	0.8638	1.6933	0.1863	17.20
Li et al. [6]	0.8747	1.6818	0.1748	12.45
Yu et al. [5]	0.8815	1.6789	0.1726	11.87
Zhang et al. [32]	0.8826	1.6781	0.1693	11.41
Sun et al. [12]	0.8830	1.6773	0.1678	10.86
Hong et al. [11]	0.8834	1.6773	0.1675	10.74
Li et al. [26]	0.8849	1.6772	0.1672	10.66
Liu et al. [7]	0.8848	1.6762	0.1667	10.45
Wei et al. [22]	0.8850	1.6760	0.1791	10.56
Li et al. [2]	0.8851	1.6770	0.1667	10.48
ASPN	0.8854	1.6762	0.1663	10.43

images in the Massachusetts data set is more than the other data sets and have a lower complexity. The second-best results are obtained from DeepGlobe → EPFL.

Fig. 10 shows the computation cost of the proposed ASPN model against the number of iterations based on the different implementation settings. As the results show, while the number of OUNs increases, the model demands more computational resources. However, still, the proposed model is more proficient than the state-of-the-art approaches. We compare the performance of ASPN with several methods on the Massachusetts road data set in Table IV and the EPFL data set in Table V. Four performance metrics are used for quantitative evaluation (PRI [41], VoI [42], GCE [43], and BDE [44]) between two segmented images (the best and the second best results are highlighted in green and red, respectively). The following conclusions have been observed from these tables. First, ASPN almost achieves the best performance in terms of all the four metrics. Second, Li *et al.* [2], Liu *et al.* [7], and Wei *et al.* [22] have better performance in the GCE and BDE and show that these three models have better boundary segmentation performance in comparison with the other models. Overall, ASPN shows excellent segmentation performance with better boundary adherence and less displacement errors with respect to the ground truth.

V. CONCLUSION

This article has presented a novel architectural model for road segmentation on RS images by using the adversarial networks and domain adaptation. A conditional generator based on FP networks was introduced, which has a wide structure to extract multilevel and multiscale features. The proposed model has several subcomponents that systematically extract the shallow and deep features on different scales, and gather various contextual information for generating the final feature maps. Hence, the model can preserve the structure of the elements in an image. The proposed model has superior performance in comparison with the other state-of-the-art approaches that use adversarial segmentation and domain adaptation. In the future, we plan to build a more efficient model to improve the segmentation accuracy not only on road but also on the other segmentation tasks whilst reducing the overall computation costs.

REFERENCES

[1] Q. Zhao, T. Sheng, Y. Wang, Z. Tang, Y. Chen, L. Cai, and H. Ling, "M2Det: A single-shot object detector based on multi-level feature pyramid network," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 9259–9266.

[2] C. Li, R. Cong, J. Hou, S. Zhang, Y. Qian, and S. Kwong, "Nested network with two-stream pyramid for salient object detection in optical remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 11, pp. 9156–9166, Nov. 2019.

[3] C. Henry, S. M. Azimi, and N. Merkle, "Road segmentation in SAR satellite images with deep fully convolutional neural networks," *IEEE Geosci. Remote Sens. Lett.*, vol. 15, no. 12, pp. 1867–1871, Dec. 2018.

[4] X. Chen, X. Lou, L. Bai, and J. Han, "Residual pyramid learning for single-shot semantic segmentation," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 7, pp. 2990–3000, Jul. 2020.

[5] B. Yu, L. Yang, and F. Chen, "Semantic segmentation for high spatial resolution remote sensing images based on convolution neural network and pyramid pooling module," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 11, no. 9, pp. 3252–3261, Sep. 2018.

[6] Y. Li, L. Guo, J. Rao, L. Xu, and S. Jin, "Road segmentation based on hybrid convolutional network for high-resolution visible remote sensing image," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 4, pp. 613–617, Apr. 2019.

[7] Y. Liu, J. Yao, X. Lu, M. Xia, X. Wang, and Y. Liu, "RoadNet: Learning to comprehensively analyze road networks in complex urban scenes from high-resolution remotely sensed images," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 4, pp. 2043–2056, Apr. 2019.

[8] J. D. Bermudez, P. N. Happ, R. Q. Feitosa, and D. A. B. Oliveira, "Synthesis of multispectral optical images from SAR/optical multitemporal data using conditional generative adversarial networks," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 8, pp. 1220–1224, Aug. 2019.

[9] B. Huang, H. Zhang, H. Song, J. Wang, and C. Song, "Unified fusion of remote-sensing imagery: Generating simultaneously high-resolution synthetic spatial–temporal–spectral Earth observations," *Remote Sens. Lett.*, vol. 4, no. 6, pp. 561–569, Jun. 2013.

[10] C. Xu and B. Zhao, "Satellite image spoofing: Creating remote sensing dataset with generative adversarial networks (short paper)," in *Proc. 10th Int. Conf. Geographic Inf. Sci. (GIScience)*. Wadern, Germany: Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018, pp. 67:1–67:6, Art. no. 67.

[11] W. Hong, Z. Wang, M. Yang, and J. Yuan, "Conditional generative adversarial network for structured domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1335–1344.

[12] W. Sun and T. Wu, "Learning spatial pyramid attentive pooling in image synthesis and image-to-image translation," 2019, *arXiv:1901.06322*. [Online]. Available: http://arxiv.org/abs/1901.06322

[13] W. Teng, N. Wang, H. Shi, Y. Liu, and J. Wang, "Classifier-constrained deep adversarial domain adaptation for cross-domain semisupervised classification in remote sensing images," *IEEE Geosci. Remote Sens. Lett.*, vol. 17, no. 5, pp. 789–793, May 2020.

[14] R. Dong, D. Xu, J. Zhao, L. Jiao, and J. An, "Sig-NMS-based faster R-CNN combining transfer learning for small target detection in VHR optical remote sensing imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 11, pp. 8534–8545, Nov. 2019.

[15] Y. Ganin *et al.*, "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2030–2096, May 2015.

[16] M. Cordts *et al.*, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 3213–3223.
[17] T. Salimans, H. Zhang, A. Radford, and D. Metaxas, "Improving GANs using optimal transport," 2018, *arXiv:1803.05573*. [Online]. Available: http://arxiv.org/abs/1803.05573
[18] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier GANs," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2642–2651.
[19] P. Shamsolmoali, M. Zareapoor, R. Wang, D. K. Jain, and J. Yang, "G-GANISR: Gradual generative adversarial network for image super resolution," *Neurocomputing*, vol. 366, pp. 140–153, Nov. 2019.
[20] M. O. Sghaier and R. Lepage, "Road extraction from very high resolution remote sensing optical images based on texture analysis and beamlet transform," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 9, no. 5, pp. 1946–1958, May 2016.
[21] G. Cheng, Y. Wang, S. Xu, H. Wang, S. Xiang, and C. Pan, "Automatic road detection and centerline extraction via cascaded end-to-end convolutional neural network," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 6, pp. 3322–3337, Jun. 2017.
[22] Y. Wei, K. Zhang, and S. Ji, "Simultaneous road surface and centerline extraction from large-scale remote sensing images using CNN-based segmentation and tracing," *IEEE Trans. Geosci. Remote Sens.*, pp. 1–13, 2020.
[23] A. Mathur, A. Isopoussu, F. Kawsar, N. B. Berthouze, and N. D. Lane, "FlexAdapt: Flexible cycle-consistent adversarial domain adaptation," in *Proc. 18th IEEE Int. Conf. Mach. Learn. Appl.*, Dec. 2019, pp. 1989–1998.
[24] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2223–2232.
[25] Q. Wang, J. Gao, and X. Li, "Weakly supervised adversarial domain adaptation for semantic segmentation in urban scenes," *IEEE Trans. Image Process.*, vol. 28, no. 9, pp. 4376–4386, Sep. 2019.
[26] Y. Li, B. Peng, L. He, K. Fan, and L. Tong, "Road segmentation of unmanned aerial vehicle remote sensing images using adversarial network with multiscale context aggregation," *IEEE J. Sel. Topics Appl. Earth Observat. Remote Sens.*, vol. 12, no. 7, pp. 2279–2287, Jul. 2019.
[27] Q. Wang, Z. Qin, F. Nie, and X. Li, "Spectral embedded adaptive neighbors clustering," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 4, pp. 1265–1271, Apr. 2019.
[28] R. Bellens, S. Gautama, L. Martinez-Fonte, W. Philips, J. C.-W. Chan, and F. Canters, "Improved classification of VHR images of urban areas using directional morphological profiles," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 10, pp. 2803–2813, Oct. 2008.
[29] J. Sivic and A. Zisserman, "Video Google: A text retrieval approach to object matching in videos," in *Proc. 9th IEEE Int. Conf. Comput. Vis.*, Oct. 2003, pp. 1470–1477.
[30] T. Mao, H. Tang, and W. Huang, "Unsupervised classification of multispectral images embedded with a segmentation of panchromatic images using localized clusters," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 11, pp. 8732–8744, Nov. 2019.
[31] X. Lu, X. Zheng, and Y. Yuan, "Remote sensing scene classification by unsupervised representation learning," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 9, pp. 5148–5157, Sep. 2017.
[32] Y. Zhang, C. Liu, M. Sun, and Y. Ou, "Pan-sharpening using an efficient bidirectional pyramid network," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 8, pp. 5549–5563, Aug. 2019.
[33] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 2117–2125.
[34] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015.
[35] J. Yang, J. Guo, H. Yue, Z. Liu, H. Hu, and K. Li, "CDnet: CNN-based cloud detection for remote sensing imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 8, pp. 6195–6211, Aug. 2019.
[36] Y. Pang, T. Wang, R. M. Anwer, F. S. Khan, and L. Shao, "Efficient featurized image pyramid network for single shot detector," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 7336–7344.
[37] E. Choi, S. Biswal, B. Malin, J. Duke, W. F. Stewart, and J. Sun, "Generating multi-label discrete patient records using generative adversarial networks," 2017, *arXiv:1703.06490*. [Online]. Available: http://arxiv.org/abs/1703.06490
[38] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.
[39] P. Shamsolmoali, M. Zareapoor, R. Wang, H. Zhou, and J. Yang, "A novel deep structure U-Net for sea-land segmentation in remote sensing images," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 12, no. 9, pp. 3219–3232, Sep. 2019.
[40] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local NASH equilibrium," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6626–6637.
[41] R. Unnikrishnan, C. Pantofaru, and M. Hebert, "Toward objective evaluation of image segmentation algorithms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 929–944, Jun. 2007.
[42] M. Meilă, "Comparing clusterings: An axiomatic view," in *Proc. 22nd Int. Conf. Mach. Learn. (ICML)*, 2005, pp. 577–584.
[43] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. 8th IEEE Int. Conf. Comput. Vis. (ICCV)*, vol. 2, Jun. 2001, pp. 416–423.
[44] J. Freixenet, X. Mñoz, D. Raba, J. Martí, and X. Cufí, "Yet another survey on image segmentation: Region and boundary information integration," in *Proc. Eur. Conf. Comput. Vis.* Berlin, Germany: Springer, 2002, pp. 408–422.
[45] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2017, pp. 2107–2116.
[46] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.
[47] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556
[48] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980