Fully Decoupled Residual ConvNet for Real-time Railway Scene Parsing of UAV Aerial Images

Lei Tong, Zhipeng Wang, Member, IEEE, Limin Jia, Member, IEEE, Yong Qin, Member, IEEE, Yanbin Wei, Huaizhi Yang, Yixuan Geng

***Abstract*- UAV-based automatic railway inspection is expected to have the potential to reform the inspection of railways. In this area, real-time railway scene parsing is quite essential. However, the limited computation resources of the UAV onboard computer pose a huge challenge for the algorithm to juggle a precise prediction with strong timeliness. Concerning this issue, this paper proposes a novel algorithm named deep fully decoupled residual convolutional network, which consists of fully decoupled residual blocks (Non-bottleneck-FDs) to deal with the dilemma between the high demand of real-time and limited resources. The residual block is constructed based on a new convolution which divides the standard convolution into three sequential convolutions to decouple the conventional operational correlations fully. Furthermore, a customized auxiliary line loss (LL) function is proposed to constrain the segmentation of railway and non-railway simultaneously without increasing the computation complexity. The proposed LL can force the predicted railway areas to concentrate in long strip areas precisely and inhibit their appearances in other impossible local areas. Subsequently, an integrated loss backpropagation strategy of the LL and cross-entropy function is presented. A comprehensive set of experiments are conducted for verification. Experiments demonstrate the superior performance of our approach with a more than 2x reduction in parameters and computation cost. Moreover, our approach also has a faster inference speed than the most existing lightweight architectures while providing comparable or higher accuracy. It is proven that our approach can reconcile the precise prediction with strong timeliness for railway scene parsing within the limitation of onboard computers. Besides, the results also imply its highest performance in terms of local details and edges of railway areas.**

***Index Terms*—Fully decoupled residual convNet, Railway scene parsing, semantic segmentation, real-time, UAV.**

I. INTRODUCTION

UNMANNED Aerial Vehicles (UAVs) has been widely used in the scene of parsing task in many fields recently. As an important auxiliary inspection approach besides manual inspection and track inspection vehicles, UAV-based automatic inspection is an important development trend in the field of high-speed railway safety operation. Instead of being limited to short maintenance time at night, UAVs can be used to conduct multi-angle and multi-period inspections on the railway equipment during the daytime with higher efficiency. Another huge advantage is that it can greatly reduce labor costs and has no impact on the train operation diagrams, so as to provide an advanced safety guarantee for railway operation. According to the existing references, UAVs has been used for rail track detection based on RGB images [1] and point clouds [2], rail surface defects detection [3], railway scene dehazing [4] and small object detection [5] in the field of railway safety operation and intelligent recognition. However, the images used in these works are mainly collected by manually controlled UAVs and few works seriously consider how to use UAVs for fully automatic and intelligent positioning, image collection, and ultimately railway key equipment detection. To achieve this, just like the effective scene parsing of the driving environment is crucial for automatic driving [6-8], real-time and efficient railway scene parsing, as a prerequisite work, is also critical for the UAV-based automatic railway inspection.

At the common flying altitude of UAVs, the image captured by an aerial camera has a wide field of view, including not only long strip-shaped railway areas but also the complex and uncertain surrounding environment along both sides of the railway. In the railway area, there are many key facilities and equipment that support the daily operation of the railway system. The normal operation of these facilities and equipment has a direct and important impact on the safe operation of the railway. Therefore, utilizing the onboard computer to accurately perceive the location of railway areas from the real-time video stream is foundational work for UAVs to carry out fully automatic railway inspections. At the same time, as can be easily understood, the railway surrounding environment also has a crucial impact on the safety of railway operations. Therefore, the automatic high-precision and real-time scene parsing of both the railway area and the surrounding environment is of the same great significance for UAV-based automatic railway inspection.

Manuscript received xx xx, 2021; revised xx xx, 2021; accepted xxx xx, 2021. Date of publication xx xx, 2022; date of current version xx xx, 2022.

This research is supported by National Natural Science Foundation of China (Nos. 91738301, 61803022 and 61833002) and Research project of Beijing Shanghai High Speed Railway company: Research on key technologies of UAV inspection for Beijing Shanghai high-speed railway infrastructure (No. I20D00010). (Corresponding authors: Zhipeng Wang and Limin Jia)

L. Tong, Z. Wang, L. Jia, Y. Qin, and Y. Geng are with State Key Laboratory of Railway Traffic Control and Safety and Key Laboratory of Railway Industry of Proactive Safety and Risk Control, Beijing Jiaotong University, Beijing 100044, China (e-mail: 20114070@bjtu.edu.cn; zpwang@bjtu.edu.cn; lmjia@bjtu.edu.cn; yqin@bjtu.edu.cn; 17114233@bjtu.edu.cn;).

Y. Wei are with 93787 army, Beijing 100044, China (email: kallenlucky@qq.com).

H. Yang are with Beijing-Shanghai High Speed Railway Co.,Ltd., Beijing 100044, China (email: gtyhz_007@126.com).

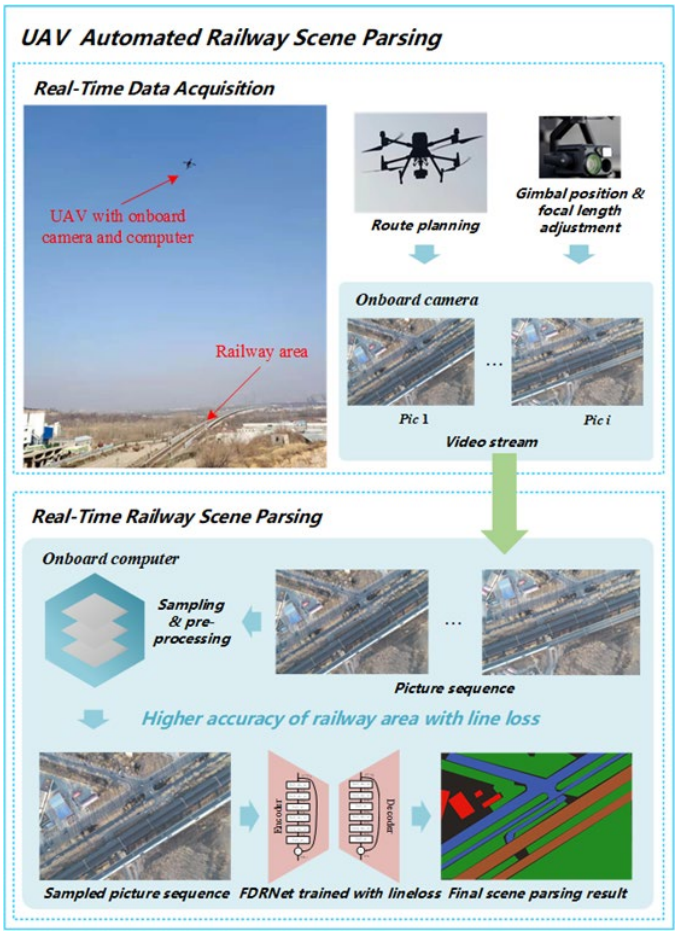


Fig. 1 UAV-based automatic railway scene parsing framework.

Nevertheless, the onboard computer has very limited storage space, power supply, and computation resources. Therefore, it is fairly necessary to develop specialized real-time scene parsing algorithms for UAV-based images. We urgently expect a solution with a small model size, fewer parameters, and fast calculation to meet this specific need. Concerning this issue, we propose a fully decoupled residual convolutional network (FDRNet), which is composed of fully decoupled residual blocks referred to as Non-bottleneck-FDs which could extremely reduce the number of parameters and computation. This proposed architecture is a typical encoder-decoder network mainly stacked with Non-bottleneck-FDs. Usually, the high-speed railway normally stretches for thousands of miles, by contrast, the length of the railway that can be photographed by the UAVs in each photo is very limited. It is often the case that the railway area extends directly to both sides of the image and is distributed in regular strips. However, among the approaches that can be used universally for scene parsing, there is no good way to make use of this excellent characteristic of UAV railway scene images. Especially, images of the railway scene captured by UAVs, in which railway areas always appear as long strips with regular edges of parallel lines, are quite different from those captured from inspection vehicles or railway surveillance [9]. In order to make full use of this characteristic of the UAV aerial images of the railway scene, we also propose a customized auxiliary line loss function for FDRNet to pursue better segmentation of railway areas non-railway areas simultaneously. As a result, an onboard UAV-based automatic railway scene parsing framework is proposed, as shown in Fig. 1. We have demonstrated the effectiveness of our architecture together with the customized auxiliary line loss function by a set of comprehensive experiments.

The contributions of this paper are summarized as follows:

(1) In view of the limited computation resources of the UAV onboard computer, this paper proposes a novel algorithm named deep fully decoupled residual convolutional network, which consists of fully decoupled residual blocks (Non-bottleneck-FDs) to deal with the dilemma between the high demand of real-time and limited resources.

(2) To make full use of the characteristics of the railway areas in UAV images for the sake of their high-precision parsing, a customized auxiliary line loss (LL) function is proposed to constrain the segmentation of railway and non-railway simultaneously without increasing the computation complexity.

This paper is organized as follows: Section II reviews the related works to real-time railway scene parsing; Section III presents the whole architecture of our approach in detail; Section IV gives the results of a series of comprehensive experiments and makes full analysis and discussion; Section V concludes this paper.

II. Related Works

Scene parsing, also referred to as semantic segmentation, means to classify each pixel in an image, which is a quite challenging visual task. In the past few years, scene parsing has made great progress with deep learnings. Through adapting the existing classification convolutional networks such as VGG [10], Long et al. first proposed a kind of fully convolutional network FCN [11] to tackle the vision task of semantic segmentation in an unprecedented end-to-end way. Yet due to its overmuch downsampling to the feature representations, FCN fails to acquire top accuracy results. Despite its additional skip connections for layer fusing to refine the pixel outputs, this does not make a big difference. For the sake of better enhancement, SegNet [12] considers executing finer upsampling with the help of indices of corresponding max-pooling modules in the encoder. And Deeplab models [13-15] exploited conditional random fields (CRF) and atrous convolution (also called dilated convolution) to pursue richer contextual feature representations as well as refinement of the output. Other top-accuracy architectures like PSPNet [8] proposed a pyramid pooling module to obtain global multi-scale representation and higher quality scene parsing results.

Despite the high accuracy of these state-of-the-art approaches, there still exists a huge gap between the state-of-the-art and realistic applications and deployment in fields like autonomous driving vehicles or UAV-based intelligent applications. The toughest obstacle is that the current advanced

networks can hardly simultaneously meet the realistic requirements of high accuracy, low latency, and low consumption. Hence it has become an urgent task to develop high-precision architectures as well as to meet the constraints of computing resources. In recent years, researchers have attempted to develop real-time networks.

On the one hand, based on the same depthwise separable convolution, Xception [16] and MobileNet [17] have been developed as well-performing and lightweight networks respectively. Although both of them are built for classification tasks, the idea of separating and decoupling standard convolutions to make the network lightweight is embedded in them, which is to separate the cross-channel correlations and spatial correlations. On the other hand, ERFNet [6] by Eduardo et al. proposed a novel layer that uses residual connections and factorized convolutions to achieve a remarkable trade-off between accuracy and efficiency on the basis of absorbing the theory about factorized convolution in [18-20] and residual designs concerning both non-bottleneck and bottleneck in ResNet [21]. However, this decomposition only involves the factorization of spatial correlations, which means turning a 2D convolution into sequential 1D convolutions. Besides, ERFNet is also required to reduce the number of parameters and model size in order to satisfy the constraints of computing resources of the onboard computer. Other works like ENet [22] adapted from ResNet to the segmentation task also suffer from the low accuracy despite its high efficiency. SQ [7] and ICNet [23] remain to be improved in terms of time consumption and real-time capacity. The performance of inference speed of BiSeNet [24] and DFANet [25] is not bad but they still suffer the accuracy in spite of their excellent results in their reports. STDC [26] and DDRNet [27] proposed recently do well both in inference forward time and prediction accuracy, but their model size still needs to be further reduced.

To our best knowledge, real-time railway scene parsing based on UAV aerial images is rarely discussed by now. The above architectures only consider this task as a discrete pixel classification problem that cannot take the inherent geometric positions of different pixels in railways and non-railway areas into account. Inspired by the idea of convolution decomposition in ERFNet and Xception, we propose a novel fully decoupled residual design. Combined with our proposed customized line loss function, we attempt to resolve the shortcomings of previous works and achieve qualified real-time railway scene parsing with the proposed formulation.

III. FDRNet Architecture

In this section, we present our FDRNet for real-time railway scene parsing with the balance of accuracy, speed, and resource consumption, along with the proposed line loss function customized for narrow strip railway areas. This network mainly focuses on the problem inherently existing in the traditional efficiency-limited residual block which is mostly adopted in the state of the art classification, object detection [28, 29], and segmentation [13, 22] tasks. Aiming at breaking through this kind of limitation, we further restructure the residual block proposed in ResNet [21] and ERFNet [6] and achieve high accuracy while maintaining low time cost and satisfying limited resources constraints. Additionally, we also propose a customized auxiliary line loss function for higher precision of railway area, which can restrict the predicted railway area concentrating in long strip regions to a greater extent.

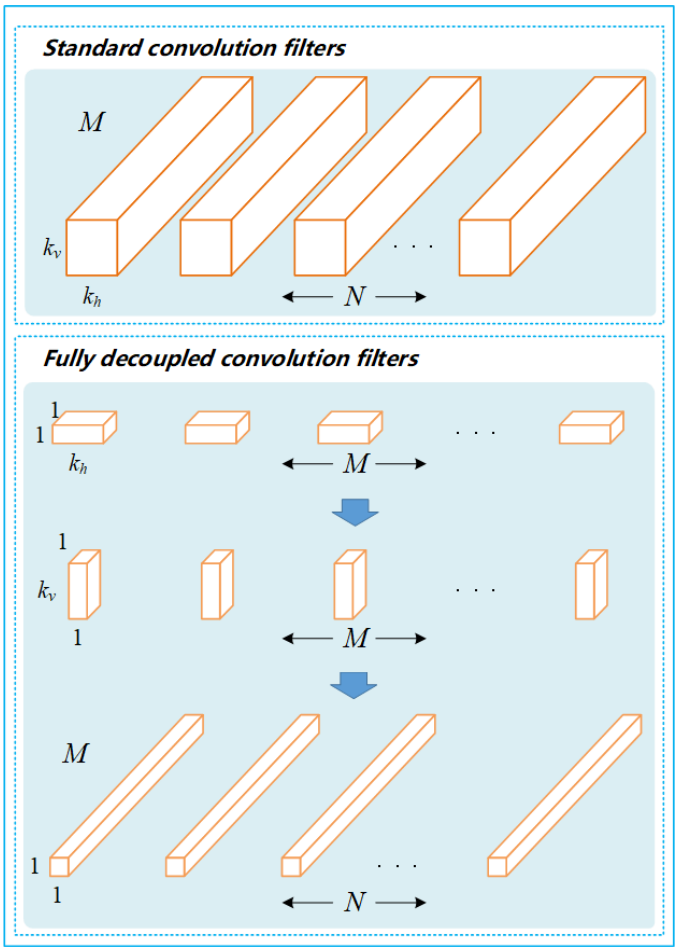


Fig. 2. Standard convolution filters and fully decoupled convolution filters.

A. Fully Decoupled Residual Block

A basic idea to propose this residual block is further decoupling of standard convolutions, which means a great reduction in parameters and calculations. That is, on the basis of keeping the basic operational correlations of convolution unchanged, the parameter quantity can be greatly reduced to avoid too much time consumption on calculation and resource occupation. The proposed fully decoupled convolution includes two aspects of operational correlations decoupling: (1) the decoupling of cross-channel operational correlations and spatial correlations; (2) the decoupling of horizontal and vertical spatial operational correlations. The correlations we talk about later in the article refer specifically to operational correlations existing in the convolution process.

Prior to this work, Xception [16], as the extreme version of

Inceptionv3 [30], and MobileNet [17] have made use of the first kind of decoupling to achieve a rather more efficient classification. However, we need to point out that their decoupling mode is not thorough enough. From the perspective of the 3D property of convolution filters, there are still two kinds of spatial correlations that fail to be decoupled. Besides, ERFNet has made a good exploration on the decoupling of spatial correlations, decomposing a 2D convolution into two successive 1D convolutions. Based on the two residual modules of ResNet, good performances have been made to reduce the number of parameters while maintaining high accuracy. However, the basic building blocks proposed in ERFNet did not perform separation of cross-channel operations and spatial operations.

Based on the above analysis, we first make the following hypothesis: In effect, the two kinds of coupling correlations mode in the convolutional networks can be entirely decoupled, that is, (1) the cross-channel correlations and spatial correlations in the feature maps can be completely decoupled; (2) furthermore, the two kinds of spatial correlations (horizontal spatial correlations and vertical spatial correlations) in the feature map can also be completely decoupled. As shown in Fig. 2, we compare the filters of the fully decoupled convolution and a standard convolution. Fully decoupled convolution decomposes the standard convolution into three sequential steps: horizontal 1D depthwise convolution, vertical 1D depthwise convolution, and cross-channel pointwise convolution. Among them, the *M* convolution kernels in the first two kinds of depthwise convolutions of different spatial dimensions correspond to the M channels of the input feature map one-to-one. The final pointwise convolution is a special case of ordinary standard convolution with a kernel size of 1x1. It mainly completes the establishment of the cross-channel correlations mapping relationship in the convolution process and can transform the channels of input feature maps from *M* to *N*. Based on such a fully decoupled convolution, we can easily get a brand-new non-bottleneck residual module, which will be described in detail later. We have compared the number of parameters for the standard convolution, the 1D convolution used in ERFNet, the convolution used in Xception and our fully decoupled convolution in this paper.

For the standard convolution, the calculation formula for the number of parameters is as follows:

$$k_v \times k_h \times M \times N = M\left(k_v \cdot k_h\right)N \tag{1}$$

For the 1D convolution proposed in ERFNet, the calculation formula of the number of parameters is as follows:

$$k_v \times 1 \times M \times N + k_h \times 1 \times M \times N = M\left(k_v + k_h\right)N \tag{2}$$

For the convolution used in Xception, the calculation formula for the number of parameters (need to be further confirmed) is as follows:

$$k_v \times k_h \times M + 1 \times 1 \times M \times N = M\left(k_v \times k_h + N\right) \tag{3}$$

or as follows:

$$1 \times 1 \times M \times N + k_v \times k_h \times N = \left(M + k_v \times k_h\right)N \tag{4}$$

The difference between these two forms is whether the convolution module in Xception should be 1x1 convolution followed by depthwise convolution or depthwise convolution followed by 1x1 convolution [16]. For the fully decoupled convolution proposed in this paper, the calculation formula for the number of parameters is:

$$k_v \times M + k_h \times M + N \times M = M\left(k_v + k_h + N\right) \tag{5}$$

We take the convolution with a kernel size of 3x3 as an example. The standard convolution has almost nine times the number of parameters of our fully decoupled convolution.

As proved in [18], any 2D filter can be represented by a combination of 1D filters. Let $W \in \mathbb{R}^{M \times k_1 \times k_2 \times N}$ represent the weights of a general 2D convolutional layer, where M is the number of input feature maps, N is the number of output feature maps and $k_1 \times k_2$ represents the size of each kernel in the filters (typically $k_1 \equiv k_2 \equiv k$). Suppose $b \in \mathbb{R}^N$ be the vector denoting the bias and $f^i \in \mathbb{R}^{k_1 \times k_2}$ be the *i*-th kernel in the convolutional layer. On this basis, [19] has proved that f^i can be rewritten as a linear combination of 1D filters:

$$f^i = \sum_{k=1}^{K} \sigma_k^i v_k^i \left(\overline{h}_k^i\right)^T \tag{6}$$

where v_k^i and $\overline{h}_k^i$ are vectors with dimension d and σ_k^i is a scalar denoting the *k*-th weight of this combination. [18] has pointed out each convolutional layer can be decomposed with 1D filters with an additional non-linearity $\varphi(\cdot)$ included in between. This way, the i-th feature map of the output a_i^1 can be formulated as the following function of the input feature maps a^0:

$$a_i^1 = \varphi\left(b_i^h + \sum_{l=1}^{L} \overline{h}_{il}^T * \left[\varphi\left(b_l^v + \sum_{m=1}^{M} \overline{v}_{lm} * a_m^0\right)\right]\right) \tag{7}$$

where L represents the number of feature maps (or the channels of internal filters) in the intermediate layer, $\varphi(\cdot)$ can be implemented with ReLU [31] or PReLU [32] and * means to execute a convolution. Yet we can notice that spatial correlations and cross-channel correlations in the above convolution mode are still not decoupled. For a fully decoupled convolution mode denoted as presented as Fig. 2, the formula can be reformulated as follows:

$$a_i^1 = \varphi\left(b_i^p + \sum_{m=1}^{M} \overline{p}_{im} * \left[\varphi\left(b_m^h + \overline{h}_m^T * \left[\varphi\left(b_m^v + \overline{v}_m * a_m^0\right)\right]\right)\right]\right) \tag{8}$$

Since the kernel $\overline{p}_{im}$ represents a 1x1 convolution which has only one scalar parameter, the formulation can be rewritten as:

$$a_i^1 = \varphi\left(b_i^p + \sum_{m=1}^{M} p_{im}\left[\varphi\left(b_m^h + \overline{h}_m^T * \left[\varphi\left(b_m^v + \overline{v}_m * a_m^0\right)\right]\right)\right]\right) \tag{9}$$

where p_{im} is the only parameter in the kernel $\overline{p}_{im}$.

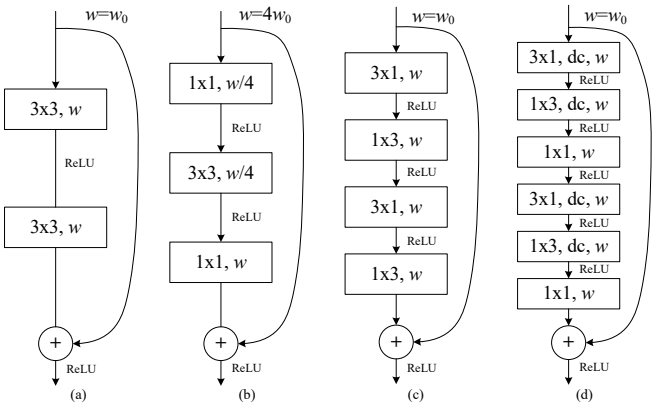


Fig. 3. Depiction of the residual layers originally proposed in [21] and [6] and our proposed residual design. (a) Non-bottleneck. (b) Bottleneck. (c) Non-bottleneck-1D. (d) Non-bottleneck-FD.Non-bottleneck-fully-decoupled (Non-bt-FD). "*w*" represents the number of channels input to the layer, internally reduced by 4 in the bottleneck design (b). "w0" denotes a specific constant value. "dc" in (d) means depthwise convolution.

We further propose our fully decoupled residual module by utilizing this decomposition which takes advantage of fully decoupled convolutional layers. Prior to this work, the original residual modules, i.e. bottleneck and non-bottleneck versions, are proposed in ResNet, as depicted in Fig. 3. (a) and (b). Considering the possible more accuracy gain with the non-bottleneck version and remaining degradation sufferings with bottleneck design [10, 33, 34], ERFNet modifies the non-bottleneck residual module with proposed 1D factorization to accelerate and reduce the parameters of the original non-bottleneck layer, referred to as non-bottleneck-1D as shown in Fig. 3 (c). Here we further modify the non-bottleneck-1D residual module with our proposed correlations fully decoupled convolution to gain more reduction in the number of parameters and time cost, as depicted in Fig.3 (d). Our fully decoupled residual layer, dubbed non-bottleneck-FD, consists of two correlations fully decoupled convolution and identity map link. We change the 1D convolution in the non-bottleneck-1D into two 1D depthwise convolutions which only consider spatial correlations and append an additional 1x1 convolution (also called pointwise convolution) to realize final cross-channel correlations of feature maps. In Fig. 3, "$k_1 \times k_2$" represents the kernel size used in the specific convolution module. "*w*" represents the number of channels input to the layer while "w_0" denotes a specific constant value. "dc" means the depthwise convolution. Note that ReLU nonlinear activations are added to every convolution.

TABLE I
COMPARISON OF FOUR KINDS OF RESIDUAL LAYERS.

Residual Block	No.-FM	Parameters	GFLOPs
Non-bottleneck	64	74.1 K	19.43
Bottleneck	256	70.8 K	18.56
Non-bottleneck-1D	64	49.7 K	13.02
Non-bottleneck-FD	64	9.6 K	2.52

We compare these four kinds of residual modules by using their parameters and calculations, as listed in Table I. For a fair comparison, 4x times more channels are given to the bottleneck module design because of its internal 4x channel reduction. These four designs are tested under the input size of 512x512. It is illustrated that the proposed non-bottleneck-FD requires the least parameters and minimal calculation.

B. Architecture Design

We suppose to design compact but efficient network architecture, as presented in Table II. Since the network parameters have been greatly reduced, which leads to a degradation of network performance inevitably, we can consider expanding the network scale to recoup the loss. Meanwhile, it can be also noticed that there are at least two more steps (no counting other additional batch normalization and following nonlinear activation layers) in the proposed non-bottleneck-FD which in itself will deepen the network and therefore increase the forward time slightly compared to the other ones.

The expansion of network scale can be conducted in two directions, one is to deepen the network, and the other is to widen the network. We attempt to do both kinds of design and experiment, and it turned out that widening the network is a more optimized direction suitable for the current deep learning framework PyTorch according to our empirical observation. Empirically, it is analyzed that the increase of convolution steps slows down the speed of the network with the possibility that PyTorch is more sensitive to the depth of networks than width.

TABLE II
ARCHITECTURE OF FDRNET. **No.-FM** MEANS THE NUMBER OF OUTPUT FEATURE MAPS AND **Size-FM** MEANS SIZE OF OUTPUT FEATURE MAPS. C REPRESENTS THE NUMBER OF PREDEFINED CLASSES.

	Layer	Blocks	No.-FM	Size-FM
ENCODER	1	**Downsampler block**	16	256x256
	2	**Downsampler block**	64	128x128
	3-7	**5 x Non-bt-FD**	64	128x128
	8	**Downsampler block**	256	64x64
	9	**Non-bt-FD** (dilated 2)	256	64x64
	10	**Non-bt-FD** (dilated 4)	256	64x64
	11	**Non-bt-FD** (dilated 8)	256	64x64
	12	**Non-bt-FD** (dilated 2)	256	64x64
	13	**Non-bt-FD** (dilated 4)	256	64x64
	14	**Non-bt-FD** (dilated 8)	256	64x64
DECODER	15	**Deconvolution** (upsampling)	64	128x128
	16	**Non-bt-FD**	64	128x128
	17	**Deconvolution** (upsampling)	16	256x256
	18	**Non-bt-FD**	16	256x256
	19	**Deconvolution** (upsampling)	C	512x512

In our architecture, the encoder segment is composed of layers from 1 to 14 and the decoder segment is composed of layers from 15 to 19. Inspired by ERFNet, we design this wider convolutional architecture, adopting the same downsampler block in layers 1, 2, and 8, which performs down-sampling with both results of max-pooling and a single 3x3 convolution with a stride of 2 concatenated to capture more abundant features. Some dilated convolutions [35] are interleaved into the second correlations decoupled convolution to gain more contextual and

global information. Here we abandon the convolutional layers with a dilated rate of 16 to avoid both little gains in semantic context and depth increasing. Additionally, the dropout [36], with an empirical rate of 0.05 instead of 0.3 in ERFNet, is also included in our architecture as a regularization term to yield optimal feature representations. For the upsampling step, three successive transposed convolutions denoted as layers 15, 17, and 19, are employed to enlarge the resolution of feature maps to the original size of the input image.

C. Auxiliary Line Loss Function

The railway area is the key area of interest for UAV-based automatic inspection. The accurate prediction of this area plays an important role in future detection work, such as fastener detection, track detection, and track plate detection. We propose a novel line loss (LL) function to improve the accuracy of the class railway in the task of railway scene parsing. The LL function focuses on the excellent characteristics of a high-speed railway which is relatively straight in a long distance. Generally, for the highway, line designers always deliberately make it bend to prevent the driver's visual fatigue. Distinguished from the concept of the deliberate turnings of highways, the longer the straight line of high-speed railways is, the better. Therefore, railways far more exceed highways in terms of straightness. Noticed that in the local area of the railway covered by the UAV remote sensing image, the railway area is often quite straight in many cases.

Traditional cross-entropy (CE) loss function tackles this task as a classification problem for each discrete pixel of a whole image, ignoring the inherent relationship across pixels. It is noticed that pixels in the railway region can always be shaped like a long strip area. Hence, the closer the pixel is to the centerline of the strip railway area, the more likely it is to be part of the railway; the farther the pixel is from the centerline, the less likely it is to belong to the railway area. To take full advantage of this idea, we present the following LL loss function, which can avoid the discrete classification problem of CE loss function across pixels from different regions while taking all pixels of railway area in one or two strip-shaped regions of an image into consideration. The proposed LL loss function can largely correct the error of the wrong classification of pixels in the railway area, which can be intrinsic to the traditional loss function. In the following part, we will give some basic definitions first, and finally introduce our LL loss function and its formula.

1) Normalized Coordinate System

Traditionally, we establish a pixel-based coordinate system with a pixel as a unit length. But when the image needs to be scaled, this coordinate system cannot satisfy the invariance of the distance between two points at the corresponding position in the image. To deal with this issue, we establish a normalized coordinate system. This coordinate system takes the width and height of the entire image as unit lengths. For an image with a size of $w_0 \times h_0$, if the traditional coordinates of a pixel are (w, h), then the normalized coordinates of the pixel are $(w/w_0,\ h/h_0)$. Under this formulation, the Euclidean distance $d(p_1, p_2)$ between the pixels $p_1: (w_1, h_1)$ and $p_2: (w_2, h_2)$ can be calculated as follows:

$$d(p_1, p_2) = \sqrt{(w_1 - w_2)^2 / {w_0}^2 + (h_1 - h_2)^2 / {h_0}^2} \qquad (10)$$

It is obvious that the normalized coordinate system has such properties: (1) the distance between two pixels is invariant to the scaling transformation of an image; (2) the distance between two pixels in the normalized coordinate system is $\sqrt{2}$ at most. The uniqueness of the DoM distribution of an image we define later in this article is precisely based on the excellent properties of the normalized coordinate system.

2) Basic assumption

The high-speed railway normally stretches for thousands of miles, by contrast, the length of the railway that can be photographed by the UAVs in each photo is very limited. Meanwhile, unlike highways, the turning radiuses of railways are very large, which means the curvatures are almost imperceptible from the UAV images. Therefore, it is often the case that the railway area extends directly to both sides of the image and is distributed in regular strips. With reasonable flight parameter settings, even if this type of line turn is encountered, the railway area can still appear as long straight strips in the relatively small field of view that can be perceived by the UAV image. So we give the following assumptions.

Assumption 1: If there exists a railway area in a UAV-based image, the edges of the area are two lines in parallel.

Assumption 2: If there exist two railway areas in a UAV-based image, they are always parallel to each other.

3) Degree of Membership

Next, we define the pixels' degrees of membership (DoM) to railway areas, which are ultimately used in the definition of the LL function.

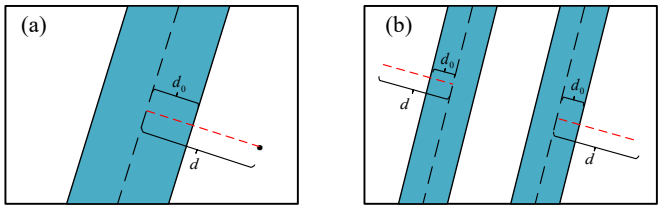


Fig. 4. Common railway areas and their center lines. (a) The centerline and distances for a single railway area. (b) Centerlines and distances for double railway areas.

Case 1: there exists a single railway area in the image as shown in Fig. 4 (a). Assume that the distance from the pixel p to the centerline l is d, and the distance from the pixels on the edges of the railway area to the centerline l is d_0.

Case 2: there exist two or more strip railway areas in the image as shown in Fig. 4 (b). Supposing the distance from the pixel p to the centerline l_k is d_k, then it is decided that the pixel p belongs to the railway area $\alpha = \arg\min_k \{d_k\}$ where

$k \in \{1, 2, ...\}$, where k means the k-th railway area. Here, we suppose that the distance from the pixel p to the centerline l_α is d, and the distance from the pixels on the edges of the α-th railway area to the centerline l_α is d_0.

Combining the above two situations, it can be defined that the DoM to railway areas for pixel p, denoted as $1/\lambda$, satisfies the following formula:

$$\lambda = d/d_0 \tag{11}$$

In this way, it can be concluded that the DoMs of the pixels in the non-railway area are all less than 1, and the pixels in the railway area are all greater than 1. We expect that the probability of pixels in the non-railway area predicted as railway (FP case) and pixels in the railway area predicted as non-railway (FN case) to be as small as possible consistently in the process of model training.

At the same time, it must be noted that, theoretically, the DoM distribution of an image is unique in the above normalized coordinate system. Consequently, for any scale transformation of an image, the pixels' DoMs to the railway areas obtained in this coordinate system always remains the same, which explains why we need to calculate the distance between pixels in such a coordinate system. For *Case* 1 and *Case* 2 discussed above, Fig. 5 provides each of them with an example and corresponding DoM spatial distribution heat map, which gives us an intuitive feeling of the definition of DoM. In the heat maps presented in Fig. 5, λ, i.e. reciprocal of the DoM $1/\lambda$, is adopted for the purpose of clarity of presentation. DoM is a distance-based measure, so it can be observed clearly that DoM values are symmetrically distributed along both sides of the railway area(s).

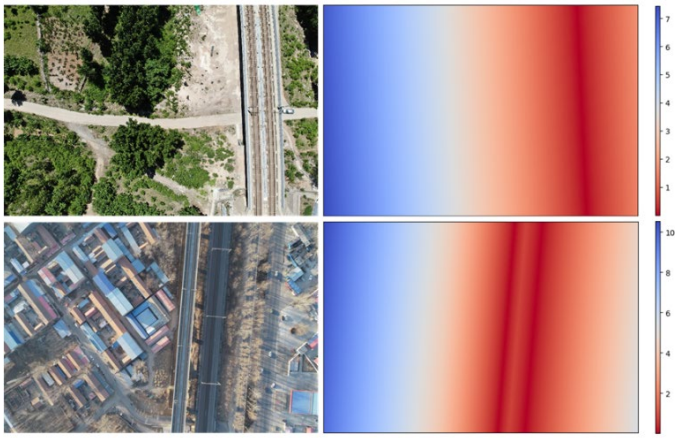


Fig. 5. The DoM spatial distribution heat maps for both situations of single railway area and double railway areas, in which λ (not $1/\lambda$) is adopted for the sake of clarity of presentation. The first row shows the DoM spatial distribution for the case of a single railway area and the second row denotes the DoM spatial distribution for the case of double railway areas.

4) Line Loss and Integrated Loss Function

Semantic segmentation is a pixel-wise dense prediction task, that is, classification is performed at each pixel. Our final purpose is to restrain both false classification pixel cases, i.e. FP and FN cases. A natural idea is to reduce the degree of probability membership of the FP case pixels while increasing the degree of probability membership of the FN case pixels. The membership here refers to the membership to the railway area. Consequently, we give the definition of line loss function as follows.

Suppose that the sets of pixels corresponding to railway and non-railway areas in an image are P_r and P_n respectively, where $|P_r| = N$ and $|P_n| = M$. For $p_i \in P_r$ and $p_j \in P_n$, suppose their DoMs are $1/\lambda_i$ and $1/\lambda_j$ respectively. With feature maps of the last layer (also called class probes) of CNN networks, probabilities that classify each pixel can be calculated using the softmax function. Assume that the probabilities that pixel p_i and p_j predicted as class railway are f_i and f_j respectively. Obviously, we expect f_i to be larger and larger until it is close to 1, while f_j to be smaller and smaller until it is close to 0 during training. In the end, our railway area line loss function can be formulated as:

$$lineloss = \frac{\frac{1}{M}\sum_{j=1}^{M}\left[f_j \cdot \lambda_j + (1-f_j)\cdot 1\right]}{\frac{1}{N}\sum_{i=1}^{N}\left[(1-f_i)\cdot \lambda_i + f_i \cdot 1\right]} - 1 \tag{12}$$

The formula indicates that if the line loss is to be reduced, f_i must be made larger and f_j smaller. If and only if $f_i = 1$ and $f_j = 0$, the LL function reaches the ideal minimum value 0. We also point out here that LL function is only suitable for binary classification of railway and non-railway, and cannot be adopted for multi-class classification tasks. Therefore, trained networks can only distinguish between railway and non-railway when only our LL function is used in the training process. Therefore, we train the network using the CE and LL loss functions simultaneously to deal with the multi-class task we face. With these two loss functions combined, CE and LL are each assigned a coefficient to form an integrated loss function, as presented below:

$$L = (1-\alpha)L_{CE} + \alpha L_{LL} \tag{13}$$

where L_{CE} represents CE loss function, L_{LL} represents LL loss function, and α denotes the ratio coefficient of LL. In this work, α is experimented with different values varying from 0 to 0.9 at an interval of 0.1 for optimal determination. More details are illustrated in part B of Section IV.

IV. Experiments

A. Dataset and General Setup

We have constructed a UAV-based railway scene parsing dataset with five labeled classes (i.e. background, railway, building, vegetation, and road), which is annotated at the pixel level. Some working photos of the UAV-based data acquisition are shown in Fig. 6. All remote sensing images captured in our

dataset are collected from three locations in Beijing-Shanghai high-speed railway: areas A, B, and C of Langfang section, as shown in Fig. 7. To demonstrate the performance of different models, 3000 images from areas A and B are used to construct the training data (2700) and validation data (300), while 300 images from area C are used for the test. Here, area C is a completely different place from areas A and B in order to avoid the possible high similarity of images between training data and test data. The proposed models and other baselines are trained on the training set without using any data in the validation and test part. All experimental results are analyzed by using the most commonly used Intersection over Union (IoU) metric:

$$IoU = \frac{TP}{TP + FP + FN} \tag{14}$$

Where TP, FP, and FN represent the numbers of pixels predicted as true positive, false positive, and false negative respectively. The means of IoUs (mIoU) of all classes are calculated as the whole evaluation of an entire prediction:

$$mIoU = \frac{1}{C}\sum_{c=1}^{C} IoU_c \tag{15}$$

In this paper, our proposed model and all baselines are trained with a batch size of 2 due to limited calculation resources of a single NVIDIA RTX 2060 6G GPU card. Adam optimizer is employed to perform stochastic gradient descent with the same momentum of 0.9, weight decay of $2e^{-4}$, and initial learning rate of $5e^{-4}$ as in ERFNet. In the experiments, we have trained all networks for 100 epochs and set a consistent random seed in the code implementation so that all networks can be trained with a fixed picture sequence to ensure the repetition of our works. The deep learning framework Pytorch is adopted to implement all deep networks modules in this paper.

Fig. 6. Some working photos of the UAV-based data acquisition.

Fig. 7. Image collection locations. (a) Area A in Langfang. (b) Area B in Langfang. (c) Area C in Langfang.

B. Integration of Loss Functions

According to empirical experiments, we have found that our proposed line loss (LL) function is much stronger in constraining networks to predict railway areas precisely than the cross-entropy (CE) loss function. However, LL function is only able to conduct binary classification on each pixel. Therefore, we should not only utilize LL function to constrain the training process of networks but also have to adopt the CE loss function for multi-class classification. Hence, how to combine CE and LL functions appropriately is a crucial problem in terms of constraining the training process on both railway and non-railway areas simultaneously. Concerning this issue, we have performed quite a different kind of training strategy which executes backpropagation with integrated loss at every single training step compared to the conventional training process. For this strategy, we have chosen different proportions of LL (line loss) varying from 0 to 0.9 at an interval of 0.1.

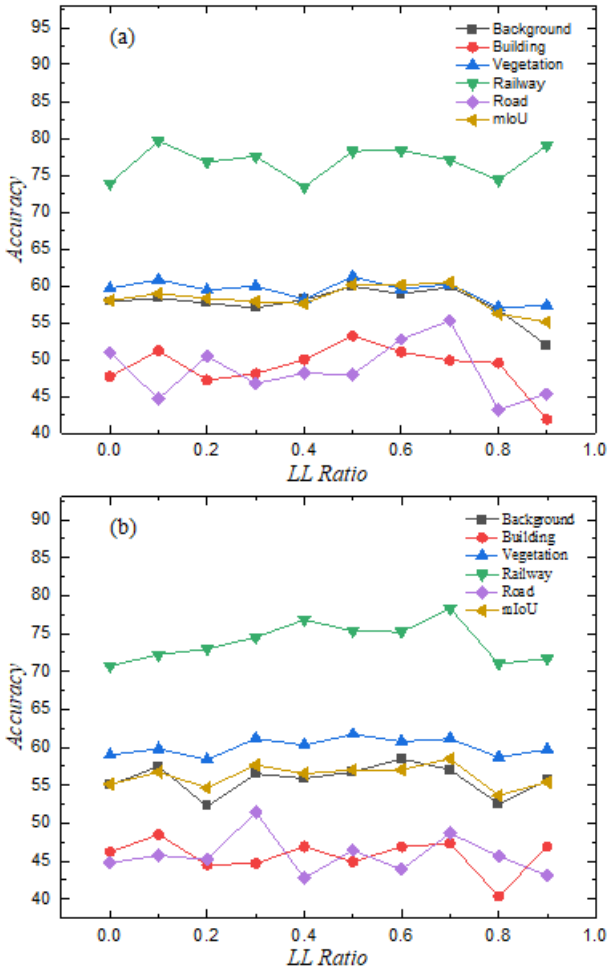


Fig. 8. Evaluation results on the test dataset using an integrated loss backpropagation strategy. (a) Results with FDRNet. (b) Results with ERFNet [6]. Both (a) and (b) consistently show that when the appropriate line loss ratio is taken, the proposed LL function can simultaneously improve the accuracy of the class railway and mIoU of the overall classes.

As shown in Fig. 8 (a) and (b), in terms of the accuracy of the railway area and mIoU of the overall classes, both the proposed FDRNet and ERFNet [6] consistently indicate that when the appropriate line loss ratio is taken, the line loss can simultaneously improve their accuracy performance. And it's revealed that both of their mIoUs reach their maximums when $\alpha = 0.7$.

Furthermore, we depicted the FDRNet's accuracy trends of all classes with LL function ratio varying from 0 to 0.9 at an

interval of 0.1 as shown in Table III. It is also illustrated the maximum accuracy increments (*Max Inc.*) of all classes and their accuracy increments when $\alpha = 0.7$ (*Inc.* α = 0.7). In the accuracy trend diagram of all classes in Table III, the points where $\alpha = 0$ and $\alpha = 0.7$ are clearly marked with bold black dots respectively. It can be concluded that the LL function can refine not only the prediction of railway areas but also the prediction of non-railway areas. As can be easily understood, when the prediction accuracy of railway area decreases, it means that more non-railway pixels are predicted as railway (FP case) or more railway pixels are predicted as non-railway (FN case), which eventually leads to the decreasing of the prediction accuracy of non-railway area.

Table III shows the accuracy growth of different classes of pixels. It can be seen that the class railway harvests the largest accuracy increment of 7.58% and the class road also gains a maximum increment of 6.66% because of its relatively strong linear characteristics. In general, the mIoU achieves a remarkable accuracy increment of 3.36%. More quantitative comparison details about accuracy between the original FDRNet and the one trained with integrated loss strategy under the best LL ratio ($\alpha = 0.7$) are listed in Table IV. It is illustrated that the accuracies of all classes rise to higher levels. The accuracies of the background, building, vegetation, railway, road increase from 54.98% to 57.04%, 46.20% to 47.34%, 58.98% to 61.12%, 70.72% to 78.30% and 44.77% to 48.67%, respectively. As a result, the mIoU increases from 55.13% to 58.49%. Especially, the railway class obtains the largest accuracy increment.

TABLE III
THE PROPOSED FDRNET'S ACCURACY PERFORMANCE OF ALL CLASSES ON THE TEST SET WITH LL FUNCTION RATIO VARYING FROM 0 TO 0.9 AT AN INTERVAL OF 0.1, WHERE "*Acc.*" MEANS ACCURACY AND "*Inc.*" MEANS INCREMENT ON ACCURACY.

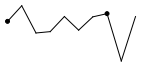
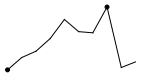
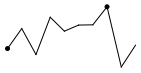
Class	*Acc. Trends*	*Max Inc.*	*Inc.* α = 0.7
background		3.49	2.06
building		2.3	1.14
vegetation		2.74	2.14
road		6.66	3.9
railway		7.58	7.58
mIoU		3.36	3.36

TABLE IV
THE QUANTITATIVE COMPARISON DETAILS ABOUT ACCURACY BETWEEN THE ORIGINAL FDRNET AND THE ONE TRAINED WITH INTEGRATED LOSS STRATEGY UNDER THE BEST LL FUNCTION RATIO, WHICH IS TAKER AS α = 0.7. THE ACCURACY OF ALL CLASSES RISES TO A HIGHER LEVEL CONSISTENTLY AND THE CLASS RAILWAY OBTAINS THE HIGHEST ACCURACY GAIN.

Network	*LL Ratio*	*Background*	*Building*	*Vegetation*	*Railway*	*Road*	*mIoU*
FDRNet	α = 0	54.98	46.20	58.98	70.72	44.77	55.13
FDRNet	α = 0.7	57.04	47.34	61.12	**78.30**	48.67	**58.49**

TABLE V
PERFORMANCE COMPARISON TO BASELINES. "**R. IoU**" MEANS IOU OF CLASS RAILWAY. "**Params**" MEANS THE NUMBER OF PARAMETERS. "**FLOPs**" ARE CALCULATED WITH INPUT OF 512x512, INDICATING THE NUMBER OF FLOATING POINT OPERATIONS. "**Model size**" IS CALCULATED USING 32-BIT FLOATING POINT DATA. "scratch" MEANS TRAINING THE MODEL FROM SCRATCH WITHOUT PRETRAINED WIGHTS LOADED.

Network	**Mode**	**R. IoU**	**mIoU**	**Params**	**FLOPs**	**Model size**
SegNet [12]	scratch	77.09	52.66	29.45 M	161.01 G	112.44 M
SQ [7]	scratch	74.18	55.83	26.37 M	104.89 G	61.97 M
ERFNet [6]	scratch	73.90	58.04	2.06 M	14.79 G	8.01 M
ICNet [23]	scratch	72.93	54.93	28.29 M	36.92 G	108.27 M
BiSeNet-18 [24]	scratch	59.51	43.43	12.80 M	13.03 G	48.94 M
BiSeNet-34 [24]	scratch	63.52	43.67	22.90 M	22.71 G	87.56 M
BiSeNet-50 [24]	scratch	66.43	47.41	27.98 M	26.40 G	107.09 M
DFANet-A [25]	scratch	58.43	48.99	2.02 M	6.95 G	8.30 M
DFANet-B [25]	scratch	56.17	48.58	1.17 M	3.38 G	5.07 M
STDC-1 [26]	scratch	78.75	57.75	12.04 M	15.55 G	46.15 M
STDC-2 [26]	scratch	77.35	54.83	16.07 M	22.16 G	61.63 M
DDRNet-23 [27]	scratch	77.73	57.95	20.15 M	17.95 G	77.11 M
DDRNet-23-slim [27]	scratch	75.19	57.32	20.15 M	17.95 G	77.11 M
DDRNet-39 [27]	scratch	76.36	58.33	32.06 M	34.03 G	122.66 M
FDRNet	scratch	70.72	55.13	1.15 M	8.07 G	4.55 M
FDRNet	scratch with LL	**78.30**	**58.49**	1.15 M	8.07 G	4.55 M
FDRNet	pretrained, with LL	**80.99**	**58.82**	1.15 M	8.07 G	4.55 M

TABLE VI
COMPARISON OF INFERENCE TIME ON A SINGLE NVIDIA JETSON TX2 EMBEDDED DEVICE.

Model	NVIDIA Jetson TX2 For Inference											
	ms	fps	ms	fps	ms	fps	ms	fps	ms	fps	ms	fps
	480x320		768x512		1280x720		512x256		1024x512		2048x1024	
SegNet	322	3.1	805	1.2	1937	0.5	266	3.8	1073	0.9	-	-
SQ	190	5.3	499	2.0	1155	0.9	159	6.3	643	1.6	2625	0.4
ERFNet	102	9.8	245	4.1	569	1.8	90	11.1	320	3.1	1250	0.8
ICNet	183	5.5	406	2.5	928	1.1	145	6.9	516	1.9	2035	0.5
BiSeNet-18	53	18.9	105	9.5	240	4.2	41	24.4	134	7.5	525	1.9
BiSeNet-34	78	12.8	157	6.4	362	2.8	64	15.6	200	5.0	832	1.2
BiSeNet-50	125	8.0	284	3.5	666	1.5	107	9.3	364	2.7	1380	0.7
DFANet-A	94	10.6	207	4.8	485	2.1	90	11.1	268	3.7	1072	0.9
DFANet-B	89	11.2	135	7.4	313	3.2	88	11.4	176	5.7	691	1.4
STDC-1	57	17.5	105	9.5	234	4.3	42	23.8	135	7.4	516	1.9
STDC-2	71	14.1	136	7.4	299	3.3	56	17.9	174	5.7	650	1.5
DDRNet-23	66	15.2	137	7.3	319	3.1	53	18.9	173	5.8	680	1.5
DDRNet-23-slim	65	15.4	137	7.3	318	3.1	53	18.9	171	5.8	678	1.5
DDRNet-39	105	9.5	225	4.4	524	1.9	84	11.9	287	3.5	1120	0.9
FDRNet	92	**10.9**	231	4.3	537	1.9	78	**12.8**	303	3.3	1197	0.8

TABLE VII
COMPARISON OF INFERENCE TIME ON A SINGLE NVIDIA GEFORCE RTX 2060 CARD.

Model	NVIDIA GEFORCE RTX 2060 For Inference											
	ms	fps	ms	fps	ms	fps	ms	fps	ms	fps	ms	fps
	480x320		768x512		1280x720		512x256		1024x512		2048x1024	
SegNet	40	25.0	88	11.4	172	5.8	34	29.4	100	10.0	-	-
SQ	24	41.7	49	20.4	96	10.4	22	45.5	63	15.9	184	5.4
ERFNet	14	71.4	25	40.0	48	20.8	13	76.9	32	31.3	93	10.8
ICNet	27	37.0	43	23.3	81	12.3	28	35.7	56	17.9	136	7.4
BiSeNet-18	10	100.0	15	66.7	26	38.5	10	100.0	17	58.8	52	19.2
BiSeNet-34	13	76.9	20	50.0	39	25.6	14	71.4	24	41.7	78	12.8
BiSeNet-50	17	58.8	29	34.5	60	16.7	17	58.8	37	27.0	103	9.7
DFANet-A	24	41.7	27	37.0	45	22.2	25	40.0	30	33.3	88	11.4
DFANet-B	23	43.5	24	41.7	33	30.3	26	38.5	26	38.5	64	15.6
STDC-1	12	83.3	17	58.8	31	32.3	12	83.3	20	50.0	59	16.9
STDC-2	15	66.7	20	50.0	36	27.8	13	76.9	26	38.5	73	13.7
DDRNet-23	12	83.3	19	52.6	36	27.8	12	83.3	23	43.5	70	14.3
DDRNet-23-slim	12	83.3	18	55.6	34	29.4	14	71.4	22	45.5	68	14.7
DDRNet-39	18	55.6	29	34.5	56	17.9	16	62.5	35	28.6	99	10.1
FDRNet	12	**83.3**	24	41.7	49	20.4	11	**90.9**	31	32.3	97	10.3

We conclude the advantages of the proposed LL function as follows: (1) It can largely improve the accuracy of both railway area and non-railway area with integrated loss strategy, and ultimately increase the overall accuracy. (2) According to the exact definition of the LL function, the model trained with LL function can make the predicted railway area in the image concentrate in long strip areas, while inhibiting the appearance of the railway area in other impossible positions. (3) The LL function has stronger interpretability for the segmentation of railway areas.

C. Comprehensive Comparisons

To demonstrate the effectiveness of our proposed method, we have made detailed and comprehensive comparisons with eight popular algorithms, i.e. SegNet [12], SQ [7], ERFNet [6], ICNet [23], BiSeNet [23], DFANet [25], STDC [26], and DDRNet [27], in terms of accuracy (R. IoU and mIoU), the number of parameters (Params), computation (FLOPs), and model size, as shown in Table V. FLOPs refers to the number of float-point operations, which is usually adopted as a metric to evaluate the computation complexity of CNN models. This metric reflects the demand for the computational capacity of onboard computers. We compare the metrics of "FLOPs" of all networks with a size-fixed input of 512x512.

All of these models are trained with the completely same parameter setting for the purpose of fair comparison of their performance. In the Table, BiSeNet-18, BiSeNet-34 and BiSeNet-50 are backboned by ResNet-18, ResNet-34 and

ResNet-50, respectively. DFANet-A and DFANet-B are backboned by XceptionA and XceptionB, respectively. STDC-1 and STDC-2 are two versions of STDC, which differ in the backbone as well. In the same way, DDRNet-23, DDRNet-23-slim, and DDRNet-39 are three different versions of DDRNet. And the number of parameters and FLOPs of all models are calculated with the same tool on the website: *https://github.com/sovrasov/flops-counter.pytorch*.

The accuracy performance of our proposed FDRNet are listed in the last three rows of the Table V. In the very beginning, our FDRNet trained from scratch without using LL function in the training stage does not obtain satisfactory accuracy result, i.e. 70.72% of R. IoU (IoU of the railway class) and 55.13% of mIoU, when compared with SQ, ERFNet, STDC-1, DDRNet series. It is supposed that the reason might be related to the large reduction of the number of parameters by using our proposed fully decoupled residual module, which could lead to accuracy degradation inevitably, in spite of the adaption of wider layers of our networks. Under our consistent parameter settings, we did not get expected accuracy results for DFANet and BiSeNet as reported, which is even not as good as FDRNet trained from scratch.

However, once the integrated loss backpropagation strategy is utilized, our proposed method improves rapidly and outperforms all other algorithms with R. IoU of 78.30% and mIoU of 58.49%. It is also noticeable that the R. IoU increases by 7.58%. Furthermore, our proposed method can finally achieve optimal R. IoU and mIoU of 80.99% and 58.82% by adopting the pre-trained weights trained on Cityscapes [37]. Experiments have proved that our customized LL loss function and integrated strategy have great significance for the railway scene parsing. It is also fairly obvious that our architecture has the least number of parameters, computation consumptions, and the smallest model size, which is of great importance for onboard application in practice.

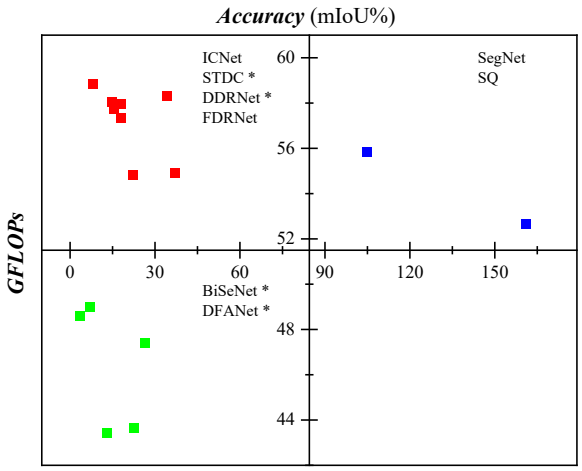


Fig. 9. The accuracy and GFLOPs performance of different models. These models can be boiled down to three classes distributed in three regions: (1) low computational cost and low accuracy, including BiSeNet * and DFANet *; (2) low computational cost but high accuracy, including ICNet, STDC *, DDRNet *, and our FDRNet; (3) high computational cost and high accuracy, including SegNet and SQ. The symbol * represents the model series.

A more in-depth analysis about the accuracy and computational cost of these models is shown in Fig. 9. These models can be boiled down to three classes distributed in three regions: (1) low computational cost and low accuracy; (2) low computational cost but high accuracy; (3) high computational cost and high accuracy. The proposed FDRNet has the highest accuracy and the smallest computational cost. The comparison among all these models also fully demonstrates that the fully decoupled convolution we proposed has great advantages in terms of parameters and computation complexity, and can maintain the basic mapping relationship of the original operational correlations existing in the standard convolution. At the same time, it also proves the excellent ability of feature extraction of our proposed basic building block Non-bottleneck-FD.

D. Computational Speed Capability

To demonstrate the computational speed capability of our method, we conducted comparative experiments under different resolutions on a single NVIDIA Jetson TX2 embedded device and a single NVIDIA GEFORCE RTX 2060 card respectively. Tables VI and VII display inference times and fps (frames per second) of our method and all other algorithms run in TX 2 and RTX 2060 respectively. The inference speed results on TX2 and RTX 2060 card here directly use the original python-based PyTorch for testing and no any C++ extensions and acceleration methods such as TensorRT are used. On this setting, we evaluate the computational speed of all models under different image aspect ratio and different image resolutions. More specifically, resolutions of 480x320, 768x512, 1280x720, 512x216, 1024x512, and 2048x1024 are used to compare the inference speeds of all models comprehensively.

We can roughly divide the inference speed of these models into three levels: the first level: inferior to FDRNet; the second level: similar to FDRNet; the third level: better than FDRNet. For the embedded device TX2, the first level is: SegNet, SQ, ICNet, BISeNet-50; the second level is: FDRNet, ERFNet, DFANet series, DDRNet-39; the third level is: BISeNet-18, BiSeNet-34, STDC series, DDRNet-23, DDRNet-23-slim. For the RTX 2060 card, the first level is: SegNet, SQ, ICNet, BiSeNet-50, DDRNet-39; the second level is: FDRNet, ERFNet, DFANet-A; the third level is: BiSegNet-18, BiSegNet-34, DFANet-B, STDC series, DDRNet-23, DDRNet-23-slim. In general, FDRNet performs well in computation speed, which can surpass most of the other models. FDRNet does have some disadvantages compared with STDC and DDRNet in real-time property, but it has absolute advantages in the number of parameters and model size. Therefore, it is still the best choice for onboard computer loaded on UAVs.

It can be seen from Table VI and Table VII that FDRNet has a greater advantage in the inference of low-resolution images. For example, in Table VII, our proposed FDRNet can achieves the fps of 83.3 and 90.9 under the resolution of 480x320 and 512x216 with RTX 2060 respectively, which is higher than

most of the other models. The inference speed of FDRNet is not lower than that of any other models except BiSenet-18 under the low resolution of 480x320. Also with the exception of BiSeNet-18, FDRNet's inference speed is significantly higher than any other model under the low resolution of 512x216. But when it comes to high-resolution areas, FDRNet is inferior to STDC series, BiSeNet-18, BiSeNet-34, DDRNet-23 and DDRNet-23-slim. Similarly, this is also reflected in Table VI.

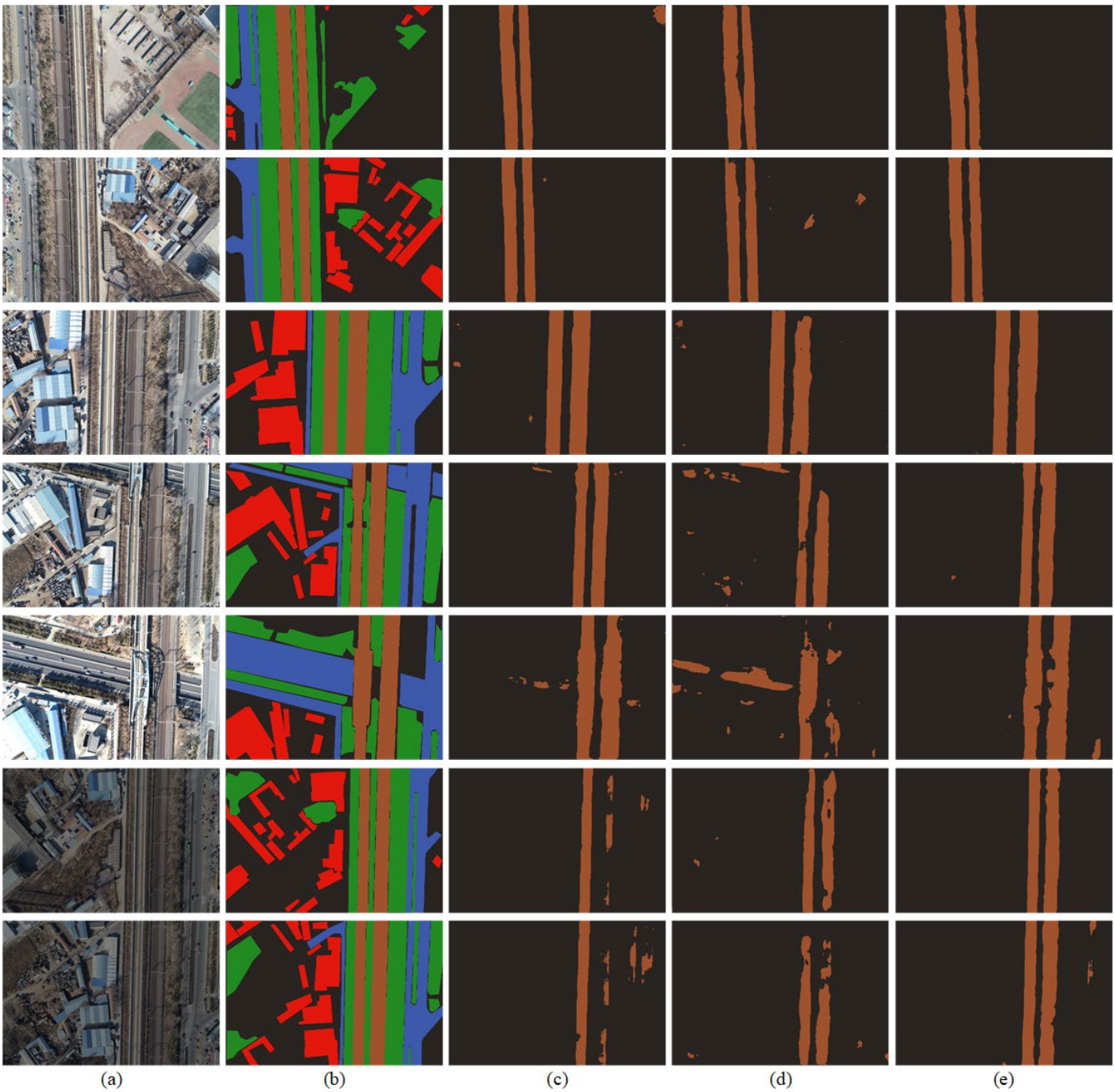


Fig. 10. Visual examples of our approach FDRNet (e) trained with integrated loss back propagation strategy compared to ERFNet [6] and DFANet-A [25]. (a) Original image. (b) Ground truth. (c) Railway segmentation of ERFNet. (d) Railway segmentation of DFANet-A. (e) Railway segmentation of FDRNet trained using integrated loss back propagation strategy with $\alpha = 0.7$ (ours).

It is worthy to point out that despite the least number of parameters and GFLOPs denoted in part C of Section IV, FDRNet actually does not obtain a computational speed as high as we expected. The number of parameters and GFLOPs of the network has been greatly reduced, but the forward inference time has not been decreased enough. The reason might be that although the decomposition of the standard convolution can greatly reduce the number of parameters and GFLOPs, it also increases the number of intermediate links, which could cause unexpected computation consumptions.

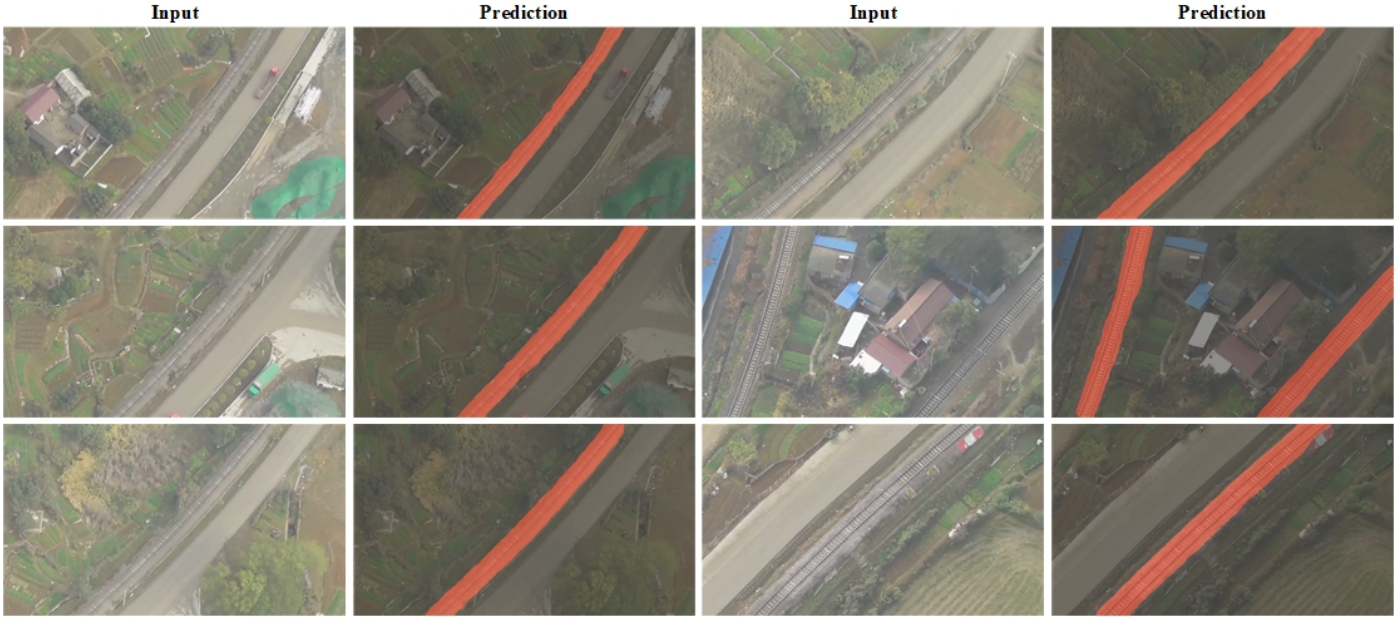


Fig. 11 Some visual examples of small-radius curves with our approach FDRNet.

E. Visual Quality of the Segmentation

We present several visual examples of our approach FDRNet (e) trained with integrated loss back propagation strategy compared to ERFNet (c) and DFANet-A (d) in Fig. 10. These results have provided strong evidence of the effectiveness of our architecture in conjunction with our customized line loss function for railway area segmentation. In the Fig. 10, the first column shows the original images and the second row shows the ground truth of those images. Several examples of railway segmentation generated by ERFNet (c), DFANet-A (d), and our FDRNet (e) are listed in the next three columns respectively. For the purpose of clearer presentation of the segmentation effect of the railway area, only class railway is included in the predicted viusal results here in Fig. 10. As illustrated, our FDRNet trained with an integrated loss backpropagation strategy can produce more precise segmentation results in terms of local details and edges than ERFNet and DFANet-A. Most of the predicted railway area is limited to the expected two long strip areas, which is exactly the intention of our customized line loss function. By comparison, although the ERFNet is well-performed to some extent, there are still a lot of coarse false positive (FP) or false negative (FN) noise pixels and regions, the effects of which are especially enlarged when dealing with images with low brightness, as denoted in the 6^{th} and 7^{th} rows of Fig. 10. This indicates our architecture is more adaptable to different lighting conditions and has more robustness in different working environments for UAV-based railway inspection.

Besides, it is necessary to point out that as a general scene parsing network architecture, our proposed FDRNet can deal with the images in which railway areas appear as any shapes, just like any other scene parsing architectures. That is determined by inherent properties of the scene parsing task which is a pixel-wise classification process of images. This can also be reflected from the fact that it can deal with the segmentation of classes vegetation and buildings (etc.) of variable shapes at the same time. Therefore, it is natural that our FDRNet can handle the scene parsing of small-radius curves in theory. We have collected ordinary railway images by a fixed-wing UAV for further verification. The fixed-wing UAV flies much higher than multi-rotor UAVs, so it is more possible to collect images of curved railway areas. As shown in Fig. 11, it can be seen that FDRNet also has an excellent segmentation performance in dealing with curved railway areas.

V. CONCLUSIONS

This paper mainly discusses real-time railway scene parsing of UAV aerial images. The contributions of this paper are summarized as follows:

(1) With regards to limited storage space and computation resource of the onboard computer and high demand of real-time property, we propose a novel deep fully decoupled residual convolutional network (FDRNet) which is composed of fully decoupled residual blocks, referred to as Non-bottleneck-FDs. The residual block is constructed based on the proposed fully decoupled convolution which decomposes the standard convolution into three sequential convolutions: horizontal 1D depthwise convolution, vertical 1D depthwise convolution, and cross-channel pointwise convolution. FDRNet needs far fewer parameters and computation consumptions compared with previous works and is able to run at over 12 fps on a Jetson TX2 embedded device and 90 fps on a single RTX 2060 card, which is faster than most popular lightweight architectures and achieves a remarkable trade-off between limited resources and real-time requirement by using the onboard computer of UAV.

(2) With regards to high precise segmentation of railway and non-railway areas, we propose a customized auxiliary loss function named as line loss (LL) function. Utilizing an

integrated loss strategy, the LL function can largely improve the accuracy of both the railway and non-railway areas and ultimately increase the overall accuracy. Especially, the LL function can make the predicted railway area concentrate in exact long strip areas and inhibit the appearance of railway areas in other impossible positions. As illustrated, our FDRNet trained with integrated loss can produce more precise segmentation results in terms of local details and edges than previous popular architectures.

In the future, we will conduct more extensive experimental verification of our method and further improve the efficiency and reduce time consumption.

REFERENCES

[1] M. Banić, A. Miltenović, M. Pavlović, and I. Ćirić, "Intelligent Machine Vision Based Railway Infrastructure Inspection and Monitoring Using Uav," *Facta Universitatis, Series: Mechanical Engineering,* vol. 17, no. 3, pp. 357-3640, 2019.

[2] S. Sahebdivani, H. Arefi, and M. Maboudi, "Rail Track Detection and Projection-Based 3D Modeling from UAV Point Cloud," *Sensors,* vol. 20, no. 18, pp. 1-15, Sep. 2020.

[3] Y. Wu, Y. Qin, Z. Wang, and L. Jia, "A UAV-Based Visual Inspection Method for Rail Surface Defects," *Applied Sciences,* vol. 8, no. 7, pp. 1028-1047, 2018.

[4] Y. Wu, Y. Qin, Z. Wang, X. Ma, and Z. Cao, "Densely pyramidal residual network for UAV-based railway images dehazing," *Neurocomputing,* vol. 371, pp. 124-136, 2020, doi: 10.1016/j.neucom.2019.06.076.

[5] Y. Li, H. Dong, H. Li, X. Zhang, B. Zhang, and Z. Xiao, "Multi - block SSD based on small object detection for UAV railway scene surveillance," *Chinese Journal of Aeronautics,* vol. 33, no. 6, pp. 1747-1755, Jun. 2020.

[6] E. Romera, J. M. Alvarez, L. M. Bergasa, and R. Arroyo, "ERFNet: Efficient Residual Factorized ConvNet for Real-Time Semantic Segmentation," *IEEE Transactions on Intelligent Transportation Systems,* vol. 19, no. 1, pp. 263-272, 2018.

[7] M. Treml *et al.*, "Speeding up Semantic Segmentation for Autonomous Driving," 2016.

[8] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid Scene Parsing Network," 2016. [Online]. Available: https://arxiv.org/abs/1612.01105.

[9] O. Zendel, M. Murschitz, M. Zeilinger, D. Steininger, S. Abbasi, and C. Beleznai, "RailSem19: A dataset for semantic rail scene understanding," in *CVPRW*, Long Beach, CA, USA, 2019, pp. 1221-1229.

[10] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, San Diego, CA, USA, 2015.

[11] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *CVPR*, 2015, pp. 431-440.

[12] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation," *IEEE Trans Pattern Anal Mach Intell,* vol. 39, no. 12, pp. 2481-2495, Dec. 2017.

[13] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 40, no. 4, pp. 834-848, 2018.

[14] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking Atrous Convolution for Semantic Image Segmentation," 2017. [Online]. Available: https://arxiv.org/abs/1706.05587.

[15] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected CRFs," in *ICLR*, 2015.

[16] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," in *CVPR*, 2017, pp. 1800-1807.

[17] A. G. Howard *et al.*, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," 2017. [Online]. Available: https://arxiv.org/abs/1704.04861.

[18] J. Alvarez and L. Petersson, "DecomposeMe: Simplifying ConvNets for End-to-End Learning," 2016. [Online]. Available: https://arxiv.org/abs/1606.05426.

[19] A. Sironi, B. Tekin, R. Rigamonti, V. Lepetit, and P. Fua, "Learning Separable Filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 37, no. 1, pp. 94-106, 2015.

[20] M. Jaderberg, A. Vedaldi, and A. Zisserman, "Speeding up convolutional neural networks with low rank expansions," in *BMVC*, Nottingham, UK, 2014.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, Las Vegas, NV, United states, 2016, pp. 770-778.

[22] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation," 2016. [Online]. Available: https://arxiv.org/abs/1606.02147.

[23] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "ICNet for Real-Time Semantic Segmentation on High-Resolution Images," in *ECCV*, Munich, Germany, 2018, pp. 418-434.

[24] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "BiSeNet: Bilateral segmentation network for real-time semantic segmentation," in *ECCV*, Munich, Germany, 2018, pp. 334-349.

[25] H. Li, P. Xiong, H. Fan, and J. Sun, "DFANet: Deep feature aggregation for real-time semantic segmentation," in *CVPR 2019*, Long Beach, CA, United states, 2019, pp. 9514-9523.

[26] M. Fan, S. Lai, J. Huang, X. Wei, and Z. Chai, "Rethinking BiSeNet For Real-time Semantic Segmentation," 2021.

[27] Y. Hong, H. Pan, W. Sun, and Y. Jia, "Deep Dual-resolution Networks for Real-time and Accurate Semantic Segmentation of Road Scenes," 2021.

[28] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," 2018. [Online]. Available: https://arxiv.org/abs/1804.02767.

[29] A. Bochkovskiy, C. Wang, and H. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," 2020. [Online]. Available: https://arxiv.org/abs/2004.10934.

[30] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," in *CVPR*, Las Vegas, NV, USA, 2016, pp. 2818-2826.

[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM,* vol. 60, no. 6, pp. 84-90, 2017.

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," presented at the ICCV, 2015.

[33] S. Zagoruyko and N. Komodakis, "Wide Residual Networks," presented at the Procedings of the British Machine Vision Conference 2016, 2016.

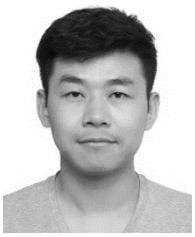
[34] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, *XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks* (Lecture Notes in Computer Science). 2016.

[35] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *ICLR 2016*, San Juan, Puerto rico, 2016.

[36] H. G. E., S. N., K. A., S. I., and S. R. R., "Improving neural networks by preventing co-adaptation of feature detectors," 2012. [Online]. Available: https://arxiv.org/abs/1207.0580.

[37] M. Cordts *et al.*, "The Cityscapes Dataset for Semantic Urban Scene Understanding," presented at the CVPR, 2016.

LEI TONG received the B.S. degree from Beijing Jiaotong University, China, in 2019. He is currently pursuing his Ph.D in the State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University, China. His research interests include deep learning, railway intelligent detection and UAV-based automatic railway inspection.



ZHIPENG WANG received the B.S. and Ph.D. degree from Northeast University and Beihang University, China, in 2008 and 2014. He is currently an associate professor in the State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University, China. His research interests include UAV-based automatic railway inspection and prognostics & health management.

LIMIN JIA is currently a Professor with the State Key Lab of Rail Traffic Control and Safety, Beijing Jiaotong University, China. His research interests include intelligent control, system safety, and fault diagnosis and their applications in a variety of fields, such as rail traffic control and safety, and transportation.

YONG QIN received the B.E. and M.E. degrees from Tongji University, Shanghai, China, in 1993 and 1996, respectively, and the Ph.D. degree from the China Academy of Railway Sciences, Beijing, China, in 1999. He is currently a Professor with the State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University, Beijing.

YANBIN WEI received the B.E. degree from Air Force Aviation University, China, in 2012. She is currently an assistant engineer in 93787 army, China. Her research interests include image processing and data analysis.

HUAIZHI YANG received the B.S. degree from Central South University, China, in 1992. He is currently a Senior Engineer in Beijing-Shanghai High Speed Railway Co.,Ltd., China. His research interests include safety guarantee and maintenance of high-speed railway infrastructures.

YIXUAN GENG is currently a Ph.D. Candidate in the State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University, China. His research interests include machine learning, fault diagnosis, and point clouds processing.