Degraded Image Semantic Segmentation With Dense-Gram Networks

Dazhou Guo, Yanting Pei, Kang Zheng, Hongkai Yu, Yuhang Lu, and Song Wang, *Senior Member, IEEE*

***Abstract*— Degraded image semantic segmentation is of great importance in autonomous driving, highway navigation systems, and many other safety-related applications and it was not systematically studied before. In general, image degradations increase the difficulty of semantic segmentation, usually leading to decreased semantic segmentation accuracy. Therefore, performance on the underlying clean images can be treated as an upper bound of degraded image semantic segmentation. While the use of supervised deep learning has substantially improved the state of the art of semantic image segmentation, the gap between the feature distribution learned using the clean images and the feature distribution learned using the degraded images poses a major obstacle in improving the degraded image semantic segmentation performance. The conventional strategies for reducing the gap include: 1) Adding image-restoration based pre-processing modules; 2) Using both clean and the degraded images for training; 3) Fine-tuning the network pre-trained on the clean image. In this paper, we propose a novel Dense-Gram Network to more effectively reduce the gap than the conventional strategies and segment degraded images. Extensive experiments demonstrate that the proposed Dense-Gram Network yields state-of-the-art semantic segmentation performance on degraded images synthesized using PASCAL VOC 2012, SUNRGBD, CamVid, and CityScapes datasets.**

***Index Terms*— Semantic segmentation, degraded images.**

I. Introduction

SEMANTIC segmentation aims to assign a categorical label to each pixel in an image [1], [2], and it plays an important role in image understanding [3]. Most existing methods assume that the input images are clean and of good quality. However, due to the complexities in the natural environment and the changes in image acquisition, e.g., optical blur, motion blur, digital noise, and natural haze, there are many cases where this assumption does not hold. In the real world, the degraded image semantic segmentation is a crucial enabler for safety-related applications, such as driving safety and precise navigation in autonomous driving and highway navigation system [4]. The recent success of deep Convolutional Neural Networks (CNNs) has made remarkable progress in pixel-level semantic segmentation tasks [1], [5]–[8]. Usually, the performance on the underlying clean images can be considered as the performance upper bound [9] on segmenting the degraded images, since the performance of segmentation networks decreases under the degraded image quality [9]. As shown by an example in Fig. 1, the train on a Gaussian blurred image is insufficiently segmented when directly employing the model pre-trained on clean images. These errors are due to the drastic changes in the appearance of objects induced by the degradation. To our best knowledge, the degraded image semantic segmentation has not been systematically studied before. In the past decades, many approaches are developed for degradation removal [10], degraded image classification [11], degraded image detection [12], and general-purpose degraded image segmentation [13], but not much work on the semantic segmentation. In this paper, we focus on developing a new approach towards degraded image semantic segmentation.

One straightforward strategy towards improving the performance of degraded image semantic segmentation is to remove the degradation effects by adding image-restoration based pre-processing. However, when the degradation degree is high, the image-restoration based pre-processing usually cannot completely restore the degraded image to its clean counterpart and may introduce additional noise to the restored images [14]. Besides, in CNN based approaches, the image-restoration based pre-processing is not integrated into the segmentation network, which also affects the segmentation performance.

Most CNN based approaches seek to gain the robustness against the degradation effects by brutal-forcedly augmenting the training dataset – using both clean and degraded images for training [15]–[17]. Specifically, a degraded image can be considered as the composition of its underlying clean image and certain additive degradation effects [18], where a degradation effect is treated as a type of image texture that does not necessarily depend on the location of semantic objects [19], [20]. Knowing that 1) the conventional network layers are not specifically tailored to effectively capture image texture [19]–[21] and 2) the supervised CNNs are task-driven frameworks, without the specific goal of capturing the image texture, the degradation effects cannot be properly addressed.

Manuscript received February 14, 2019; revised June 16, 2019; accepted July 30, 2019. Date of publication August 26, 2019; date of current version October 9, 2019. This work was supported in part by NSFC under Grant 61672376 and Grant U1803264. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Xiaochun Cao. *(Corresponding author: Song Wang.)*

D. Guo, K. Zheng, and Y. Lu are with the Department of Computer Science and Engineering, University of South Carolina, Columbia, SC 29201 USA (e-mail: guo22@email.sc.edu; zheng37@email.sc.edu; yuhang@email.sc.edu).

Y. Pei is with the School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China (e-mail: 15112073@bjtu.edu.cn).

H. Yu is with the Department of Computer Science, University of Texas-Rio Grande Valley, Edinburg, TX 78539 USA (e-mail: hongkai.yu@utrgv.edu).

S. Wang is with the Department of Computer Science and Engineering, University of South Carolina, Columbia, SC 29201 USA, and also with the College of Intelligence and Computing, Tianjin University, Tianjin 300072, China (e-mail: songwang@cec.sc.edu).

Digital Object Identifier 10.1109/TIP.2019.2936111

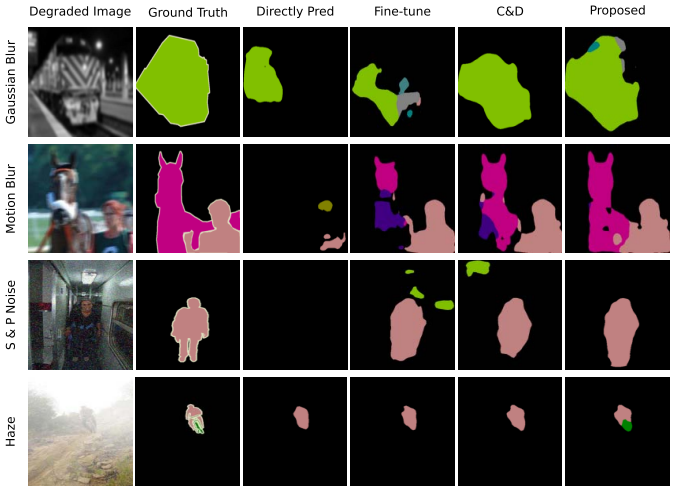


Fig. 1. Examples of semantic segmentation results on degraded images. From left to right, the six columns are degraded images, ground-truth segmentation, segmentation using the model trained on the clean images, segmentation using network fine-tuned on the degraded images, segmentation using network trained on both clean and degraded images (C&D), and segmentation using the proposed method trained on the degraded images.

Besides, using both clean and degraded images for training becomes increasingly time-consuming in training as more training images are added [8], [22].

With the emergence of image segmentation benchmarks [23]–[26] and the high activity of advancement in the field of semantic segmentation, more and more models pre-trained on the clean images are made publicly available. Network fine-tuning on the existing models [1], [5], [27], [28] becomes a popular strategy towards improving the degraded image semantic segmentation performance. By design, the features learned in the higher/deeper layers of the network are semantic/task-related [29], [30]. We expect that the distribution of the higher-layer features learned using clean images should be similar to the distribution of the higher-layer features fine-tuned using degraded images [29]–[32]. However, when fine-tuning the network using the degraded images, catastrophically forgetting the learned features of the clean images is inevitable [33]. This causes an increased gap in feature distributions of higher layers [30]. We observe that this gap in feature distributions poses a major obstacle in improving the segmentation performance of degraded images.

In this paper, we propose a novel approach to effectively reduce the gap and segment degraded images. The proposed network, as illustrated in Fig. 2, consists of two identical networks – source and target networks. Both source and target networks are initialized using the model pre-trained on the clean image while only fixing the parameters of source network during training. The feature distribution in higher-layers is quantified using the Gram matrix [30]. Exploiting the capability of the Gram matrix in capturing the image textures, the Gram matrices from the source network can be considered as the image texture of the clean images, and the Gram matrices from the target network can be considered as the image texture of the degraded image. This way, matching the Gram matrices between the source and target networks can simultaneously 1) reduce the gap in feature distributions and 2) minimize the bias induced by degradation effects.

To enhance the transferability between the source and target networks, we match the Gram matrices in a dense-interweaving manner. Within the same convolutional block, the Gram matrix of the feature maps of a layer in one network is matched to Gram matrices of the feature maps of all layers of the same block in the other network. Because of this dense-interweaving manner, we refer to our approach as Dense-Gram Network (DGN). During deployment, we only use the trained target network to segment unseen degraded images, such that no extra time or cost is added to the segmentation network.

We evaluate the proposed DGN using synthetic degraded images generated based on four benchmark datasets: PASCAL VOC 2012 [23], SUNRGBD [24], CamVid [25], and CityScapes [26]. The proposed DGN is evaluated on different state-of-the-art segmentation networks and significantly outperforms the baselines when the degradation degree is high. To sum up, the main contributions of this paper are: **1)** We systematically study the problem of the degraded image semantic segmentation. **2)** We observe that gap between feature distribution learned using the clean images and the feature distribution learned using the degraded images poses a major obstacle in improving the segmentation performance of the degraded images. **3)** We propose a novel DGN to segment degraded images and achieve substantially improved semantic segmentation performance on degraded images without adding extra time or cost during the deployment.

II. Related Work

For degraded image semantic segmentation, one naive strategy is to directly deploy the model, which is previously trained using clean images, to segment the degraded images. It is not surprising that the segmentation networks trained on clean images perform poorly on the degraded images. By design, CNN is a data-driven framework [17]. Differences in the representations of semantic objects between the clean and degraded images would cause shifts in feature distributions [34], resulting in a decrease in segmentation performance [3]. Another strategy towards improving the segmentation performance of the degraded images is to first restore the degraded image to a clearer image such that human vision can better identify object and structure details present in the image. And then, we deploy the model, which is previously trained using the clean images, to segment the degraded images [34]. However, when the degradation degree is high, the image-restoration based pre-processing usually cannot completely remove those degradation effects, such that the restored images are different from their clean counterparts in both color and texture [14]. As demonstrated in Section IV-D, using the restored images for validation, most existing image-restoration methods only improve the degraded image semantic segmentation performance by a small margin. Recently, fine-tuning the network using the degraded images is a popular strategy for improving the segmentation performance [1], [5], [6], [35]. However, fine-tuning based methods depend on the assumption that the segmentation networks can learn invariant representations that

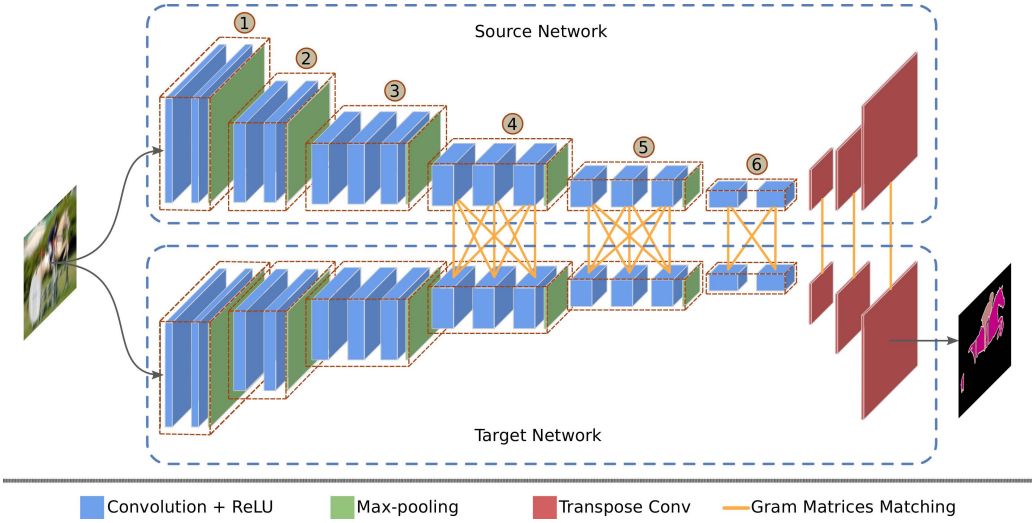


Fig. 2. An illustration of the proposed DGN architecture. The baseline network is FCN8s with VGG16 network backbone. The circled numbers denote the convolutional blocks.

are transferable between the clean and degraded images [30]. The differences between the clean and degraded images pose a bottleneck to the feature transferability and hinder the segmentation network from further improvements.

Also related to our work are the researches on domain adaptation [31], [33], style transfer [20], and knowledge distillation [36]. Kirkpatrick *et al.* [33] proposed an elastic weighting consolidation approach to remember the old tasks by selectively slowing down learning on the weights for the old tasks. Li and Hoiem [31] proposed a Learning without Forgetting network, which uses only new classification-task data to train the network while preserving the original capabilities. Johnson *et al.* [20] proposed a perceptual loss to transfer the style of a fixed image to target images by matching the Gram matrices in a layer-wise manner. In contrast, the proposed DGN enhances the feature transferability by matching the Gram matrices in a dense-interweaving manner. Romero *et al.* [36] proposed a Knowledge Distillation network, where knowledge is transferred from a large network to a smaller network for efficient deployment using the original input. Unlike the earlier approaches, the proposed DGN aims to find new network parameters for the same network structure with the goal of improving the semantic segmentation performance of the degraded images. Unlike the degraded images synthesized using clean images, in practice, the real degraded images are more difficult to collect and annotate, as well as quantifying their degradation levels. With limit-sized dataset, model overfitting could be a potential issue when using real images. In [37], Zhu et al. proposed an adversarial deep structured network to help address the issue. Yet, the proposed DGN uses the synthesized images to avoid the model overfitting issue.

III. Method

In the following, we first give an overview of the proposed DGN in Section III-A. Then, we elaborate on the proposed dense-Gram loss and the proposed DGN training in Section III-B. The analysis of the proposed DGN performance is discussed in Section III-C.

A. Dense-Gram Network Overview

The architecture of the proposed DGN is based on the teacher-student networks [36], as illustrated in Fig. 2, where the source network provides "hints" for the training of the target network. The architecture of the source and target networks are identical and the parameters of both networks are initialized using the model trained on the clean images, while fixing the parameters of the source network during training. The total number of parameters in the proposed DGN is twice the number as the original network. During the network training, we only aim to train the target network. The proposed DGN is trained in an end-to-end fashion. During the network testing, we only use the trained target network for evaluation. This way, we do not add extra time or cost during the deployment. In the following section, we elaborate on the proposed dense-Gram loss and the proposed DGN training.

B. Dense-Gram Loss & Network Training

Let $\mathcal{D}_d = \{\mathbf{x}_d^{(i)}, \mathbf{y}^{(i)}\}_{i=1}^{n_d}$ denote the degraded image dataset with n_d labelled samples, where $\mathbf{x}_d$ and $\mathbf{y}$ denote the degraded images and the corresponding segmentation ground truth, respectively.

In CNNs, we define $\phi(\cdot)$ as a composite function of following consecutive operations: batch normalization, followed by rectified linear unit and a 3×3 convolution. Let $f^{(l)}(\mathbf{x}_d) = \phi^{(l)}(\phi^{(l-1)}(\cdots\phi^{(1)}(\mathbf{x}_d)))$ denote the feature maps of the l^{th} layer. Let $c^{(l)}$ denote the number of channels of the feature maps and let $m^{(l)}$ denote the size (height times width) of the feature maps. Each element in the Gram matrix g is defined as:

$$g_{i,j}^{(l)}(\mathbf{x}_d) = \sum_{k=1}^{m^{(l)}} f_{i,k}^{(l)}(\mathbf{x}_d) f_{j,k}^{(l)}(\mathbf{x}_d), \tag{1}$$

where $i, j \in \{1, \cdots, c^{(l)}\}$ indicates the i^{th} and the j^{th} channels of the feature maps, and $g^{(l)}_{i,j}(\cdot)$ is the value (at location (i, j) of the Gram matrix g) of the inner product of the i^{th} and j^{th} vectorized feature maps of the respective i^{th} and the j^{th} channels in the l^{th} layer. Since the Gram matrix captures information about which features tend to activate together [20], the texture of the image can be well represented using the Gram matrix [19].

Let $g^{(l)}_s(\mathbf{x}_d)$ and $g^{(l)}_t(\mathbf{x}_d)$ denote the Gram matrix of the source and target networks at the l^{th} layer, respectively. The distance between the Gram matrices of the source and target networks is defined as follow:

$$\delta^{(l,l)}_{\text{Gram}} = \frac{1}{4c^{(l)2}m^{(l)2}} \left|\left| g^{(l)}_s(\mathbf{x}_d) - g^{(l)}_t(\mathbf{x}_d) \right|\right|_2. \tag{2}$$

To enhance the feature transferability between the source and target networks, within the same convolutional block, the Gram matrix of the feature maps of one layer in one network is matched to Gram matrices of the feature maps of all layers of the same block in the other network. Note that, within the same convolutional block, the dimensions of the feature maps generated from different layers are the same, i.e., for the b^{th} block with L^b layers, $c^{(l)} = c, m^{(l)} = m$, $\forall l \in \{1, \cdots, L^{(b)}\}$. The dense-Gram loss of the b^{th} block is defined as:

$$\begin{aligned} \mathcal{L}^b_{\text{Gram}} &= \sum_{l=1}^{L^b} \sum_{l'=1}^{L^b} \delta^{(l,l')}_{\text{Gram}} \\ &= \frac{1}{4c^2m^2} \sum_{l=1}^{L^b} \sum_{l'=1}^{L^b} \left|\left| g^{(l)}_s(\mathbf{x}_d) - g^{(l')}_t(\mathbf{x}_d) \right|\right|_2. \end{aligned} \tag{3}$$

During the proposed DGN training, both source and target networks are initialized using the model trained on clean images, while fixing the parameters of the source network. As discussed in [31], [38], for teach-student network based designs, using the samples of new tasks and keeping the parameters of the source network (pre-trained using clean images) fixed could help the target network preserve similar classification performance of the source network. The network design of the proposed DGN is in-line with the "learning without forgetting" strategy discussed in [31] – requiring only degraded images for training while preserving the original segmentation performance on clean images. In this paper, the default DGN only uses the degraded images for training, when there is no ambiguity. In Section IV-C, we also report the segmentation performance of clean images using the DGN trained using degraded images.

For the semantic segmentation task, let $\theta_t(\cdot)$ and $\theta_s(\cdot)$ denote the target and source networks, respectively. Given the degraded image $\mathbf{x}_d$ as the input of the proposed DGN, the prediction of the target network is $\theta_t(\mathbf{x}_d)$. For the semantic segmentation task, we only use the segmentation loss of the target network for training. We follow [1] and adopt the sigmoid cross-entropy loss for optimization. The segmentation loss is defined as:

$$\mathcal{L}_{\text{Seg}} = -\mathbf{y} log(\theta_t(\mathbf{x}_d)) + (1 - \mathbf{y}) log(1 - \theta_t(\mathbf{x}_d)). \tag{4}$$

The overall training loss of the proposed DGN is the combination of weighted dense-Gram loss and the sigmoid cross-entropy loss:

$$\mathcal{L} = \mathcal{L}_{\text{Seg}} + \lambda \sum_{b=1}^{B} \mathcal{L}^b_{\text{Gram}}, \tag{5}$$

where λ is a balance coefficient. The impact of λ selection is reported in Section IV-F. The parameter B is the number of convolutional blocks with the dense-Gram matchings.

By forcing the feature distribution in the target network to be similar to the feature distribution in the source network, the dense-Gram matching can be considered as a form of regularization. Thus, the paired convolutional blocks have to be selected such that the target network is not over-regularized. Based on the experimental results, we only apply the dense-Gram matching to the convolutional blocks located in the second half of the network. As shown in Fig. 2, for FCN8s with VGG16 network backbone [1], the dense-Gram matching starts at the 4^{th} convolutional block. The impact of the block selection is discussed in Section IV-G.

C. Analysis of the Dense-Gram Networks Performance

The proposed DGN aims to train a target network $\theta_t(\cdot)$, which is guided by the source network $\theta_s(\cdot)$, to 1) reduce the gap between feature distribution learned using the clean and the feature distribution learned using degraded images, and 2) learn effective features for the target network in degraded image semantic segmentation task, i.e., $\mathbf{y} = \theta_t(\mathbf{x_d})$. We first analyze the upper bound of segmentation performance of the proposed DGN.

Statistically, let probability distributions p and q characterize the distributions of the feature maps of the source and target networks, respectively. As proved in [39], the gap between p and q can be measured using distance between the Gram matrices of the corresponding feature maps, i.e., δ_{Gram}. The gap in feature distribution vanishes if and only if $p = q$.

Let $\epsilon_d(\theta_t) = \Pr_{(\mathbf{x}_d,\mathbf{y})\sim q}\left[\theta_t(\mathbf{x}_d) \neq \mathbf{y}\right]$ and $\epsilon_d(\theta_s) = \Pr_{(\mathbf{x}_d,\mathbf{y})\sim p}\left[\theta_s(\mathbf{x}_d) \neq \mathbf{y}\right]$ be the risks of the target and source network when using the degraded images for training, respectively. Noted by the proof in [30], [40], the target risk can be bounded by

$$\epsilon_d(\theta_t) \leqslant \epsilon_d(\theta_s) + 2\delta_{\text{Gram}} + C, \tag{6}$$

where C is a constant for the risk of an ideal null hypothesis, i.e., $p = q$, for both feature distributions and the complexity of the hypothesis space.

Let $\epsilon_c(\theta'_s) = \Pr_{(\mathbf{x}_c,\mathbf{y})\sim p'}\left[\theta'_s(\mathbf{x}_c) \neq \mathbf{y}\right]$ be the risk of the source network when using the clean images for training, where $\mathbf{x}_c$ denotes the clean images. We can expect the risk of source network to be bounded by $\epsilon_c(\theta'_s) \leqslant \epsilon_d(\theta_s)$. Since the source network is fixed during training, the feature distributions of the source network remain unchanged, i.e., $p = p'$, such that $\epsilon_c(\theta'_s) = \epsilon_c(\theta_s)$. Based on Eq. (6), the risk of the target network can be bounded by

$$\begin{aligned} \epsilon_d(\theta_t) &\leqslant 2\delta_{\text{Gram}} + 2\epsilon_d(\theta_s) - \epsilon_c(\theta_s) + C \\ &= 2\delta_{\text{Gram}} + C', \end{aligned} \tag{7}$$

where $C' = 2\epsilon_d(\theta_s) - \epsilon_c(\theta_s) + C$. When using paired clean and degraded images as the input of the respective source and target networks, i.e., $\epsilon_d(\theta_s) = \epsilon_c(\theta_s)$, the upper bound of the target risk $\epsilon_d(\theta_t)$ can be further reduced to $\epsilon_d(\theta_t) \leqslant 2\delta_{\text{Gram}} + \epsilon_c(\theta_s) + C$. As can be seen, this $2\delta_{\text{Gram}} + \epsilon_c(\theta_s) + C$ is the performance upper bound of what our proposed DGN can achieve. In the later experiment, we report the performance upper bound of the proposed DGN in Section IV-C.

Secondly, we analyze the rationale of the minimization of the dense-Gram loss. During network training, the proposed DGN performs two tasks: 1) Semantic segmentation; 2) Densely-interweaving Gram matrices matching. The proposed DGN seeks to learn optimal network parameters by jointly minimizing the segmentation loss and the dense-Gram loss, such that the target risk $\epsilon_d(\theta_t)$ can be reduced. By design, minimizing the segmentation loss can effectively minimize the target risk $\epsilon_d(\theta_t)$. Furthermore, based on Eq. (7), it is δ_{Gram} and C' that affect the upper bound of the target risk. Since the source network is fixed during training, the source risk $\epsilon_d(\theta_s)$ becomes a constant. Therefore, by minimizing the dense-Gram loss, the upper bound of the target risk in the proposed DGN can be further decreased, which leads to an improved segmentation performance.

IV. Experiments

A. Datasets & Evaluation Metric

1) Datasets: The **PASCAL VOC 2012** dataset is a natural object segmentation dataset which has been the benchmark challenge for segmentation over years. The dataset consists of $11,355$ images with $8,498$ training images and $2,857$ validation images. In total, there are 21 semantic classes that are pixel-level annotated.

The **SUNRGBD** dataset is a challenging and large indoor scene segmentation benchmark dataset with both RGB images and the corresponding depth maps. The dataset consists of $5,285$ training images and $5,050$ testing images. In total, there are 37 semantic classes that are pixel-level annotated. We only use the RGB images for training and testing.

The **CamVid** dataset is a road scene segmentation benchmark dataset which is of current practical interest for various autonomous driving related problems. The dataset consists of 367 training images, and 100 validation images, and 233 testing images. In total, there are 11 semantic classes that are pixel-level annotated.

The **CityScapes** dataset is a recently released dataset for semantic urban street scene understanding. The dataset consists of $5,000$ finely pixel-level annotated images: $2,975$ training images, 500 validate images, and $1,525$ testing images. In total, there are 19 semantic classes. The resolution of the image is $1,024 \times 2,048$. In addition, $20,000$ coarsely annotated images are provided. In this paper, we only use finely annotated images for training.

2) Metric: The metric used for segmentation performance evaluation is the mean intersection over union (mIoU).

For clarification, the results reported in all Tables are evaluated using the *testing* datasets of different benchmarks. For the CityScapes dataset, the ground-truth segmentation maps for the training and validation datasets are given while the ground-truth segmentations for the testing dataset are hidden for the user. The testing results of the CityScapes dataset are obtained via on-line submissions.

B. Implementation Details

The proposed pipeline is conducted using PyTorch,[1] MatConvNet,[2] and Tensorflow[3] implementations with Intel Core i7 6700K and with Nvidia 1080Ti GPUs with mini-batch Stochastic Gradient Descent (SGD). The segmentation network is trained using SGD with momentum of 0.9, weight decay of 0.0001 and adaptive learning rates. The mini-batch size is 1. When training networks using both clean and degraded images for baseline comparison methods, we follow the same practices reported in the original papers. When fine-tuning the networks, we follow the standard procedure for the network training [31], where smaller learning rates are adopted (10^{-1} times the original rate). The initial learning rates for the FCN-8s [1], DeepLab v2 [5], DeepLab v3 Plus [28], RefineNet [6], EncNet [41], PSPNet [27], and DLA [42] are 1×10^{-11}, 1×10^{-5}, 7×10^{-5}, 5×10^{-5}, 1×10^{-4}, 1×10^{-4}, 1×10^{-3}, respectively. We use the "poly" learning rate policy [5], where the initial learning rate is multiplied by $\left(1 - \frac{iter}{max\_iter}\right)^{power}$ with $power = 0.9$ [27]. As suggested by [43], the number of training iterations is $140,000$ for all experiments.

The degradation effects include: Gaussian blur, linear motion blur, salt & pepper noise, and haze. To better demonstrate the effectiveness of the proposed DGN, we evaluate each degradation effect using five degradation degrees $\mathbf{d}$:

- The degree of the Gaussian blur is quantified by the standard deviation of the Gaussian kernel, $\mathbf{d} \in \{1, 2, 3, 4, 5\}$.
- The degree of the linear motion blur is quantified by the motion length, $\mathbf{d} \in \{5, 10, 15, 20, 25\}$.
- The degree of salt & pepper noise is quantified by the noise density, $\mathbf{d} \in \{0.02, 0.04, 0.06, 0.08, 0.10\}$.
- The degree of haze is quantified by the scattering coefficient of atmosphere, $\mathbf{d} \in \{1.5, 2.0, 2.5, 3.0, 3.5\}$.

For fair comparison, the training data is the same for each type of degraded images without adding any extra data. The only exceptions are made when 1) training the baseline networks using both clean and degraded images, and 2) training the DGN using the paired clean and degraded images. In this paper, the approaches using both clean and degraded images are denoted using "C&D" as postfix, when there is no ambiguity.

C. Comparison to Baseline Segmentation Networks

To demonstrate the effectiveness of the proposed DGN, we train and evaluate the proposed DGN in four datasets using the baseline segmentation networks with published pre-trained models. For *PASCAL VOC 2012* dataset, five baseline networks – FCN8s, DeepLab v2, RefineNet, EncNet, and DeepLab v3 Plus – are evaluated. For *SUNRGBD* dataset,

[1] https://github.com/pytorch
[2] http://www.vlfeat.org/matconvnet/
[3] https://www.tensorflow.org/

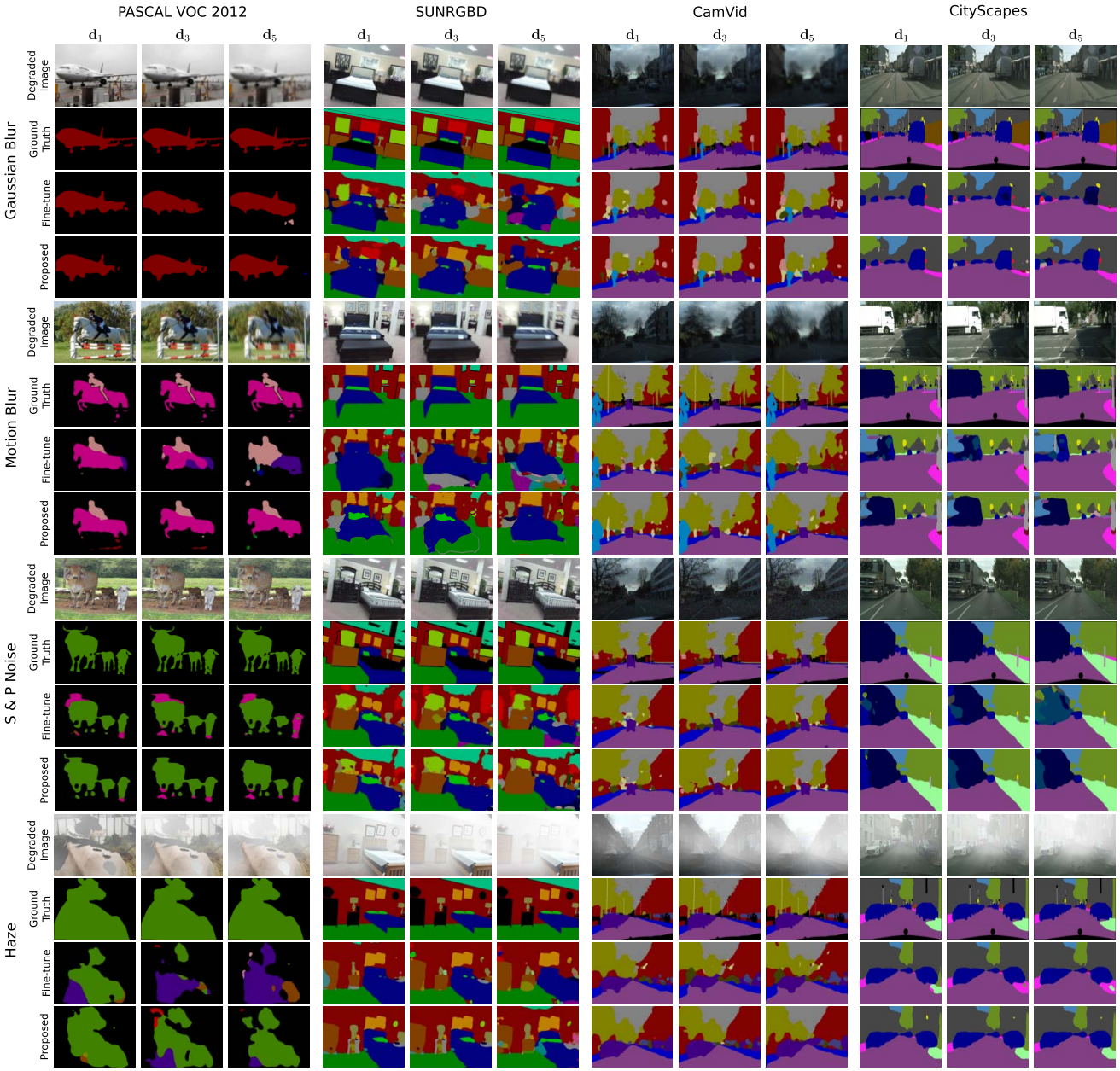


Fig. 3. Examples of semantic segmentation results on the degraded images. For each degradation effect, we select the degradation degrees of $\mathbf{d}_1$, $\mathbf{d}_3$, and $\mathbf{d}_5$ for demonstration. The baseline segmentation network is FCN8s.

three baseline networks – FCN8s, DeepLab v2, and RefineNet – are evaluated. For *CamVid* dataset, three baseline networks – FCN8s, DeepLab v2, and DLA – are evaluated. For *CityScapes* dataset, four baseline networks – FCN8s, PSPNet, DLA, DeepLab v3 Plus – are evaluated. The quantitative experiment results are shown in Table I.

Firstly, we compare the proposed DGN to the fine-tuning based counterparts. The proposed DGN achieves substantial improvements when the degradation degree is high, e.g., $\mathbf{d}_5$. Specifically, using the *PASCAL VOC 2012* testing dataset for evaluation, it shows averagely 3.2%, 3.4%, 3.5%, 3.1%, 3.7% improvements for FCN8s, DeepLab v2, RefineNet, EncNet, and DeepLab v3 Plus, respectively. Using the *SUNRGBD* testing dataset for evaluation, it shows averagely 1.9%, 2.7%, and 3.1% improvements for FCN8s, DeepLab v2, and RefineNet, respectively. Using the *CamVid* testing dataset for evaluation, it shows averagely 3.4%, 3.6%, and 3.7% improvements for FCN8s, DeepLab v2, and DLA, respectively. Using the *CityScapes* testing dataset for evaluation, it shows averagely 3.7%, 4.0%, 3.5%, 3.3% improvements for FCN8s, PSPNet, DLA, and DeepLab v3 Plus, respectively.

Secondly, we compare the proposed DGN to the baseline networks fine-tuned using both clean and degraded images. Note that the proposed DGN **only** uses the same degraded images for training without adding any extra data. Using the *PASCAL VOC 2012* testing dataset for evaluation, it shows averagely 1.2%, 1.7%, 1.6%, 1.7%, 1.7% improvements for FCN8s, DeepLab v2, RefineNet, EncNet, and DeepLab

TABLE I

THE MIOUS (IN PERCENTAGE) OF SEGMENTING DEGRADED IMAGES USING: 1) BASELINE NETWORKS FINE-TUNED USING THE DEGRADED IMAGES (*+FINE-TUNE); 2) BASELINE NETWORKS TRAINED USING BOTH CLEAN AND DEGRADED IMAGES (*+C&D); 3) DGN TRAINED USING THE DEGRADED IMAGES (*+DGN); 4) DGN TRAINED USING PAIRED CLEAN AND DEGRADED IMAGES (*+DGN+C&D). "CLEAN" DENOTES THE MIOUS ON THE CLEAN IMAGES. THE FIVE DEGRADATION DEGREES ARE DENOTED USING $\mathbf{d}_1$, $\mathbf{d}_2$, $\mathbf{d}_3$, $\mathbf{d}_4$, AND $\mathbf{d}_5$, RESPECTIVELY. THE NUMBERS WITH THE BETTER AND BEST PERFORMANCE ARE HIGHLIGHTED IN BLUE AND RED, RESPECTIVELY

PASCAL VOC 2012

	Clean	Gaussian Blur					Motion Blur					S & P Noise					Haze				
		$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$
FCN8s+fine-tune	67.2	67.0	64.1	60.7	56.4	52.8	64.8	58.4	52.1	49.8	45.4	66.3	65.5	64.6	63.4	61.6	65.6	63.9	62.2	60.5	60.4
FCN8s+C&D		67.0	65.7	64.0	60.0	55.8	65.3	59.3	54.1	50.9	47.1	66.9	66.0	65.5	63.8	63.1	66.6	65.3	64.1	62.1	62.0
FCN8s+DGN		**67.1**	**66.2**	**64.7**	**61.3**	**57.4**	**65.4**	**59.7**	**54.7**	**52.2**	**48.1**	**66.9**	**66.4**	**66.0**	**65.1**	**64.2**	**66.7**	**65.7**	**64.9**	**63.3**	**63.1**
FCN8s+DGN+C&D		**67.1**	**66.4**	**65.1**	**61.7**	**58.1**	**65.4**	**60.0**	**55.0**	**52.6**	**48.8**	**66.9**	**66.5**	**66.4**	**65.6**	**65.3**	**66.7**	**65.9**	**65.2**	**64.1**	**64.1**
DeepLab v2+fine-tune	79.7	79.6	77.1	73.6	69.1	63.8	76.9	69.8	63.7	61.2	56.3	78.2	76.8	75.5	73.1	72.3	77.1	73.9	72.5	71.1	70.4
DeepLab v2+C&D		79.4	77.5	74.5	70.5	67.3	77.6	71.2	65.9	62.6	57.5	78.9	77.6	76.6	74.3	73.3	78.5	76.2	74.7	73.0	71.5
DeepLab v2+DGN		**79.6**	**77.8**	**75.2**	**71.9**	**69.1**	**77.8**	**71.6**	**66.6**	**63.9**	**59.1**	**78.9**	**77.9**	**77.3**	**75.4**	**74.9**	**78.7**	**76.6**	**75.4**	**74.1**	**73.3**
DeepLab v2+DGN+C&D		**79.7**	**78.1**	**75.5**	**72.4**	**69.8**	**77.8**	**71.7**	**66.9**	**64.6**	**59.9**	**78.9**	**78.0**	**77.7**	**75.9**	**75.7**	**78.9**	**76.7**	**75.7**	**74.7**	**74.1**
RefineNet+fine-tune	82.4	82.3	79.4	75.4	72.6	65.8	79.1	72.1	66.3	62.9	57.8	81.4	79.2	77.9	75.9	75.2	79.9	77.2	74.8	73.7	73.4
RefineNet+C&D		82.3	79.2	77.5	73.9	70.3	80.0	73.6	68.5	65.0	58.7	82.1	80.1	79.2	77.4	76.0	81.5	79.6	77.3	75.7	74.7
RefineNet+DGN		**82.3**	**79.6**	**78.1**	**74.7**	**71.2**	**80.1**	**73.9**	**69.1**	**65.7**	**60.5**	**82.1**	**80.4**	**79.8**	**78.3**	**77.9**	**81.6**	**80.1**	**77.8**	**76.8**	**76.5**
RefineNet+DGN+C&D		**82.3**	**79.9**	**78.3**	**75.4**	**71.9**	**80.3**	**74.1**	**69.3**	**66.2**	**61.3**	**82.2**	**80.5**	**80.2**	**78.7**	**78.6**	**81.8**	**80.3**	**78.2**	**77.5**	**77.3**
EncNet+fine-tune	82.9	82.2	80.1	77.5	74.4	70.7	80.7	74.3	70.4	64.2	60.2	82.4	80.2	80.1	77.3	75.1	82.0	80.5	78.8	76.3	75.8
EncNet+C&D		82.8	80.9	77.7	75.2	71.7	81.6	76.0	70.6	66.6	62.3	82.6	81.3	80.6	78.2	77.0	82.1	80.5	78.9	77.2	76.3
EncNet+DGN		**82.8**	**81.4**	**78.4**	**76.3**	**73.6**	**81.6**	**76.4**	**70.2**	**67.8**	**63.7**	**82.8**	**81.7**	**81.3**	**79.3**	**79.0**	**82.3**	**80.9**	**79.7**	**78.3**	**78.0**
EncNet+DGN+C&D		**82.8**	**81.6**	**78.8**	**76.7**	**74.2**	**81.6**	**76.6**	**70.5**	**68.3**	**64.9**	**83.0**	**81.9**	**81.6**	**79.7**	**79.7**	**82.4**	**81.1**	**79.9**	**78.9**	**78.8**
DeepLab v3 Plus+fine-tune	89.0	88.4	86.5	82.7	78.3	73.0	85.8	79.0	72.9	70.2	65.3	87.1	86.0	84.6	82.2	81.2	86.5	83.2	81.5	79.9	79.4
DeepLab v3 Plus+C&D		88.7	86.8	83.9	79.6	77.0	86.9	80.3	75.3	72.1	66.7	88.2	86.9	86.2	83.8	82.3	87.8	85.4	84.1	82.2	81.0
DeepLab v3 Plus+DGN		**88.8**	**87.2**	**84.5**	**80.9**	**78.7**	**86.9**	**80.7**	**75.8**	**73.4**	**68.1**	**88.3**	**87.3**	**86.8**	**84.9**	**84.2**	**88.0**	**85.8**	**84.8**	**83.6**	**82.8**
DeepLab v3 Plus+DGN+C&D		**88.9**	**87.5**	**84.9**	**81.5**	**79.5**	**87.1**	**80.9**	**76.1**	**74.1**	**69.3**	**88.5**	**87.5**	**87.0**	**85.7**	**84.9**	**88.0**	**86.1**	**85.1**	**84.4**	**83.5**

SUNRGBD

	Clean	Gaussian Blur					Motion Blur					S & P Noise					Haze				
		$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$
FCN8s+fine-tune	27.4	23.7	21.9	20.5	19.6	18.2	23.0	20.8	18.8	17.5	16.1	21.0	20.6	19.8	18.7	19.0	25.5	22.5	22.2	21.5	20.4
FCN8s+C&D		24.8	23.6	22.5	20.9	19.8	23.4	21.8	19.4	17.9	16.6	21.8	21.4	20.5	19.1	18.7	25.6	23.2	22.8	21.6	20.4
FCN8s+DGN		**24.9**	**24.1**	**23.2**	**22.1**	**21.0**	**23.6**	**22.2**	**20.1**	**19.1**	**17.8**	**21.9**	**21.7**	**21.1**	**20.3**	**20.1**	**25.8**	**23.7**	**23.5**	**22.8**	**22.3**
FCN8s+DGN+C&D		**25.1**	**24.4**	**23.5**	**22.8**	**22.1**	**23.6**	**22.4**	**20.4**	**19.6**	**18.5**	**22.0**	**21.9**	**21.5**	**21.0**	**21.3**	**25.9**	**23.9**	**23.9**	**23.5**	**22.9**
DeepLab v2+fine-tune	32.1	28.3	26.7	25.4	23.7	21.9	27.8	25.4	23.1	21.8	20.4	26.3	25.5	24.6	22.9	22.6	30.8	28.1	26.9	25.8	25.1
DeepLab v2+C&D		29.5	28.6	27.6	25.4	23.7	28.4	26.4	24.0	22.4	20.9	27.3	26.4	25.2	23.7	22.8	31.2	30.0	28.7	27.5	27.2
DeepLab v2+DGN		**29.6**	**29.0**	**28.3**	**26.9**	**25.6**	**28.5**	**26.7**	**24.6**	**23.5**	**22.2**	**27.4**	**26.8**	**25.9**	**25.1**	**24.7**	**31.2**	**30.5**	**29.4**	**28.8**	**28.4**
DeepLab v2+DGN+C&D		**29.7**	**29.3**	**28.6**	**27.5**	**26.7**	**28.6**	**26.9**	**25.0**	**24.0**	**22.9**	**27.5**	**26.9**	**26.1**	**25.8**	**25.4**	**31.4**	**30.7**	**29.6**	**29.4**	**29.0**
RefineNet+fine-tune	45.7	41.9	40.2	38.9	37.1	35.6	41.2	39.3	36.2	34.9	32.5	39.1	38.6	38.0	36.4	35.8	43.2	41.1	40.2	39.2	38.1
RefineNet+C&D		43.1	42.1	41.2	39.0	37.9	41.5	40.2	37.2	35.8	33.1	40.2	39.4	39.0	37.2	36.4	43.3	42.2	41.8	40.9	40.7
RefineNet+DGN		**43.1**	**42.6**	**41.7**	**40.4**	**39.5**	**41.6**	**40.5**	**37.8**	**36.9**	**34.8**	**40.3**	**39.7**	**39.6**	**38.6**	**38.1**	**43.4**	**42.6**	**42.4**	**42.2**	**41.8**
RefineNet+DGN+C&D		**43.2**	**42.8**	**42.0**	**40.8**	**40.4**	**41.7**	**40.7**	**38.1**	**37.4**	**36.0**	**40.5**	**40.0**	**39.8**	**39.3**	**38.9**	**43.5**	**42.8**	**42.8**	**42.7**	**42.7**

CamVid

	Clean	Gaussian Blur					Motion Blur					S & P Noise					Haze				
		$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	d_5
FCN8s+fine-tune	56.7	53.0	49.1	46.2	43.7	41.7	51.6	46.2	45.2	43.7	41.5	51.7	50.0	48.4	47.3	46.2	53.6	52.3	50.7	49.6	48.9
FCN8s+C&D		54.2	51.5	48.7	47.0	45.5	54.0	48.5	46.5	43.7	42.0	51.8	50.4	49.5	48.8	47.7	53.7	52.8	51.8	50.6	49.4
FCN8s+DGN		**54.2**	**51.9**	**49.5**	**48.2**	**47.5**	**54.2**	**48.9**	**47.2**	**45.1**	**43.7**	**51.9**	**50.7**	**50.2**	**50.1**	**49.6**	**53.8**	**53.2**	**52.5**	**51.7**	**51.2**
FCN8s+DGN+C&D		**54.3**	**52.1**	**49.8**	**48.7**	**48.5**	**54.2**	**49.0**	**47.5**	**45.5**	**44.3**	**51.9**	**50.9**	**50.6**	**50.7**	**50.4**	**53.9**	**53.3**	**52.7**	**52.1**	**51.8**
DeepLab v2+fine-tune	61.6	57.4	53.9	50.3	48.1	45.7	56.5	51.3	50.7	49.2	47.1	57.1	55.2	52.9	51.7	50.4	58.8	57.0	55.2	54.2	52.6
DeepLab v2+C&D		57.9	56.4	54.1	51.6	49.2	59.0	53.7	53.1	49.7	48.5	57.4	55.8	54.4	53.5	52.1	59.0	57.8	56.7	55.2	53.2
DeepLab v2+DGN		**57.9**	**56.8**	**54.8**	**52.6**	**51.7**	**59.2**	**54.2**	**53.7**	**50.8**	**49.6**	**57.4**	**56.1**	**55.1**	**54.7**	**53.9**	**59.1**	**58.1**	**57.3**	**56.5**	**55.1**
DeepLab v2+DGN+C&D		**57.9**	**56.9**	**55.0**	**53.2**	**52.7**	**59.3**	**54.5**	**54.0**	**51.5**	**50.6**	**57.5**	**56.3**	**55.4**	**55.1**	**54.7**	**59.2**	**58.3**	**57.6**	**56.9**	**56.3**
DLA+fine-tune	66.7	62.7	58.7	56.1	53.4	51.6	61.4	55.9	55.0	53.7	51.1	61.4	60.0	58.5	56.8	55.8	63.7	62.0	60.4	59.4	58.4
DLA+C&D		64.4	61.5	59.0	57.0	55.3	64.1	58.7	56.3	53.7	51.9	61.6	60.5	59.2	58.6	57.9	63.7	62.4	61.6	60.2	58.9
DLA+DGN		**64.4**	**62.0**	**59.8**	**58.1**	**57.2**	**64.3**	**59.1**	**56.9**	**55.0**	**53.8**	**61.6**	**60.9**	**59.8**	**59.9**	**59.8**	**63.7**	**62.8**	**62.4**	**61.3**	**60.9**
DLA+DGN+C&D		**64.6**	**62.2**	**60.0**	**58.8**	**58.2**	**64.3**	**59.2**	**57.2**	**55.5**	**54.6**	**61.7**	**61.2**	**60.0**	**60.6**	**60.4**	**63.8**	**63.1**	**62.7**	**61.9**	**61.6**

CityScapes

	Clean	Gaussian Blur					Motion Blur					S & P Noise					Haze				
		$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	d_5
FCN8s+fine-tune	65.3	65.2	63.9	63.0	61.8	60.6	65.1	63.2	62.8	61.3	60.8	60.3	57.7	55.6	54.2	52.9	63.1	62.3	61.6	59.2	57.9
FCN8s+C&D		65.2	64.1	63.6	62.7	61.4	65.1	64.1	63.0	61.7	61.3	64.7	62.8	61.9	58.7	56.9	64.1	63.4	62.4	61.2	59.8
FCN8s+DGN		**65.2**	**64.5**	**64.2**	**63.8**	**63.2**	**65.2**	**64.5**	**63.7**	**63.1**	**62.7**	**64.9**	**63.2**	**62.7**	**60.1**	**59.7**	**64.2**	**63.8**	**63.2**	**62.3**	**61.4**
FCN8s+DGN+C&D		**65.3**	**64.6**	**64.5**	**64.5**	**64.0**	**65.3**	**64.6**	**64.1**	**63.7**	**63.4**	**65.0**	**63.4**	**62.9**	**60.7**	**60.5**	**64.3**	**64.1**	**63.5**	**62.7**	**62.2**
PSPNet+fine-tune	78.4	78.1	77.3	76.1	73.7	71.4	77.3	76.4	75.1	72.4	71.7	73.3	70.6	68.6	67.3	66.0	76.4	75.3	74.5	72.5	70.8
PSPNet+C&D		78.1	77.1	76.5	74.4	72.5	77.8	77.3	75.7	73.9	72.2	77.7	75.6	75.0	72.0	70.8	77.5	76.1	75.0	74.4	72.5
PSPNet+DGN		**78.2**	**77.5**	**77.0**	**75.9**	**74.6**	**78.0**	**77.7**	**76.3**	**75.9**	**74.8**	**77.8**	**76.1**	**75.5**	**73.2**	**72.5**	**77.6**	**76.5**	**75.6**	**75.5**	**74.0**
PSPNet+DGN+C&D		**78.3**	**77.7**	**77.2**	**76.5**	**75.4**	**78.2**	**77.9**	**76.7**	**76.5**	**76.0**	**77.9**	**76.3**	**75.9**	**73.7**	**73.4**	**77.7**	**76.8**	**75.9**	**76.1**	**74.9**
DLA+fine-tune	73.8	73.4	72.4	72.5	71.1	70.2	73.3	71.5	71.1	69.7	68.9	68.4	66.1	63.7	62.3	61.4	71.6	70.8	70.1	67.7	66.4
DLA+C&D		73.6	72.6	72.0	71.0	70.4	73.5	72.8	71.7	70.2	69.1	72.9	71.1	70.2	67.1	66.5	73.0	72.4	71.3	69.7	68.1
DLA+DGN		**73.6**	**73.0**	**72.7**	**72.1**	**72.0**	**73.5**	**73.2**	**72.4**	**71.6**	**70.5**	**73.0**	**71.5**	**70.7**	**68.5**	**68.3**	**73.2**	**72.7**	**71.9**	**71.0**	**70.0**
DLA+DGN+C&D		**73.7**	**73.3**	**72.9**	**72.7**	**73.1**	**73.6**	**73.5**	**72.7**	**72.1**	**71.4**	**73.1**	**71.7**	**71.1**	**69.1**	**69.3**	**73.2**	**73.0**	**72.1**	**71.6**	**70.9**
DeepLab v3 Plus+fine-tune	82.1	81.9	80.6	80.6	79.1	77.6	81.9	79.6	79.3	77.8	77.5	77.1	74.3	72.0	70.7	69.8	79.8	78.9	78.0	75.7	74.2
DeepLab v3 Plus+C&D		**82.0**	80.7	80.2	79.4	78.1	82.1	80.6	79.5	78.3	77.1	81.7	79.4	78.5	75.6	73.0	80.9	80.0	79.1	77.7	76.4
DeepLab v3 Plus+DGN		81.9	**81.2**	**80.8**	**80.6**	**79.6**	**82.1**	**81.0**	**80.2**	**79.6**	**79.1**	**81.8**	**79.9**	**79.1**	**76.8**	**75.6**	**81.0**	**80.4**	**79.6**	**79.0**	**77.8**
DeepLab v3 Plus+DGN+C&D		**82.0**	**81.3**	**81.2**	**81.0**	**80.5**	**82.1**	**81.1**	**80.6**	**80.4**	**80.2**	**81.9**	**80.2**	**79.4**	**77.4**	**76.4**	**81.1**	**80.6**	**79.8**	**79.7**	**78.5**

v3 Plus, respectively. Using the *SUNRGBD* testing dataset for evaluation, it shows averagely 1.5%, 1.6%, and 1.5% improvements for FCN8s, DeepLab v2, and RefineNet, respectively. Using the *CamVid* testing dataset for evaluation, it shows averagely 1.8%, 1.8%, and 1.9% improvements for FCN8s, DeepLab v2, and DLA, respectively. Using the *CityScapes* testing dataset for evaluation, it shows averagely 1.9%, 2.0%, 1.7%, 1.9% improvements for FCN8s, PSPNet, DLA, and DeepLab v3 Plus, respectively.

Thirdly, as discussed in Section III-C, we conduct the experiments to test the upper bound performance of the proposed DGN. We use paired clean and degraded images as the respective inputs of the source and target networks and compare the segmentation performance to the proposed DGN.

TABLE II

THE MIOUS (IN PERCENTAGE) OF SEGMENTING CLEAN IMAGES USING 1) THE FINE-TUNED BASELINE, 2) THE TRAINED DGN, 3) THE TRAINED DGN+C&D. THE FIVE DEGRADATION DEGREES ARE DENOTED BY $\mathbf{d}_1$, $\mathbf{d}_2$, $\mathbf{d}_3$, $\mathbf{d}_4$, AND $\mathbf{d}_5$, RESPECTIVELY. THE NUMBERS WITH THE BEST PERFORMANCE ARE HIGHLIGHTED IN RED

		PASCAL VOC 2012					SUNRGBD					CamVid					CityScapes				
		$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$
Gaussian Blur	FCN8s+fine-tune	66.9	64.2	60.4	56.0	52.4	23.9	22.1	20.7	19.5	18.3	53.1	49.2	46.1	43.9	41.8	64.5	62.1	61.3	59.6	57.7
	FCN8s+DGN	67.2	66.4	65.1	64.2	63.9	26.3	25.7	24.9	24.5	23.7	55.9	53.6	51.5	50.8	50.9	65.2	64.8	64.4	64.5	64.9
	FCN8s+DGN+C&D	**67.2**	**66.6**	**65.6**	**64.7**	**64.7**	**27.0**	**26.4**	**26.1**	**25.4**	**25.0**	**56.2**	**54.1**	**52.2**	**51.8**	**51.5**	**65.4**	**65.1**	**65.2**	**65.6**	**65.7**
Motion Blur	FCN8s+fine-tune	66.2	60.0	53.3	50.9	46.4	24.3	22.0	20.1	18.7	17.2	53.0	47.4	46.3	44.8	42.7	64.5	63.4	63.4	63.0	62.0
	FCN8s+DGN	67.0	61.6	56.6	54.8	51.5	26.1	24.7	22.8	22.4	21.8	56.3	51.2	49.6	48.0	47.0	65.3	64.9	64.1	64.0	64.4
	FCN8s+DGN+C&D	**67.1**	**61.5**	**57.0**	**55.2**	**51.7**	**26.7**	**25.4**	**23.9**	**23.6**	**23.0**	**56.8**	**51.8**	**50.5**	**49.2**	**48.3**	**65.5**	**65.2**	**64.7**	**64.8**	**64.5**
S & P Noise	FCN8s+fine-tune	66.1	65.5	64.4	63.3	61.0	20.8	19.9	19.6	18.6	18.5	51.5	49.7	47.9	46.9	45.8	64.7	62.5	60.9	57.6	56.4
	FCN8s+DGN	67.2	66.6	66.5	66.1	66.5	27.1	27.0	26.5	26.3	26.8	56.9	55.7	55.4	55.9	55.9	65.1	63.5	63.2	61.1	61.7
	FCN8s+DGN+C&D	**67.3**	**67.2**	**67.2**	**66.9**	**66.5**	**27.3**	**27.2**	**27.2**	**26.8**	**27.3**	**57.3**	**56.3**	**56.3**	**56.8**	**57.2**	**65.3**	**63.9**	**63.8**	**61.8**	**61.9**
Haze	FCN8s+fine-tune	66.1	64.3	62.6	60.7	60.8	25.9	22.8	22.5	21.8	20.6	54.1	52.9	51.2	50.1	49.2	64.3	63.2	61.6	60.9	59.2
	FCN8s+DGN	**67.3**	66.5	65.7	64.8	65.0	27.1	26.1	25.4	25.0	25.5	56.8	56.4	55.8	55.5	55.2	65.0	64.7	64.2	63.8	64.1
	FCN8s+DGN+C&D	67.3	**66.7**	**66.3**	**65.5**	**65.5**	**27.3**	**26.4**	**25.9**	**26.1**	**25.8**	**57.2**	**56.9**	**56.6**	**56.2**	**56.0**	**65.2**	**65.1**	**64.9**	**64.5**	**64.8**

Using the *PASCAL VOC 2012* testing dataset for evaluation, it shows averagely 0.9%, 0.8%, 0.7%, 0.8%, 0.8% additional improvements for FCN8s, DeepLab v2, RefineNet, EncNet, and DeepLab v3 Plus, respectively. Using the *SUNRGBD* testing dataset for evaluation, it shows averagely 0.9%, 0.8%, and 1.0% additional improvements for FCN8s, DeepLab v2, and RefineNet, respectively. Using the *CamVid* testing dataset for evaluation, it shows averagely 0.8%, 1.0%, and 0.8% additional improvements for FCN8s, DeepLab v2, and DLA, respectively. Using the *CityScapes* testing dataset for evaluation, it shows averagely 0.8%, 0.9%, 1.0%, 0.8% additional improvements for FCN8s, PSPNet, DLA, and DeepLab v3 Plus, respectively.

We conduct an additional experiment to evaluate the segmentation performance of the clean images using the fine-tuned baseline, the trained DGN, and the trained DGN+C&D. The experimental results are reported in Table II. In comparison to the fine-tuned baseline, the proposed DGN+C&D constantly achieves the best performance. In comparison to the baseline pre-trained and evaluated using the clean images, the DGN trained using the degraded images only decreases the performance by a small margin – by averagely 1.5%.

To better understand the relationship between the segmentation performance and the dense-Gram loss, as shown in Fig. 4, we provide a sample of training curves on *PASCAL VOC 2012* $\mathbf{d}_5$ degree Gaussian blur images. Note that we follow the same network design as the proposed DGN to only calculate the dense-Gram losses for the fine-tuning based and C&D based approaches. Using the dense-Gram loss for quantification, we observe that, when fine-tuning the network using the degraded images, the gap in the distributions of features learned using the clean and degraded images first drops but then increases along with the training iterations. The C&D based approach reduces the gap, but not very significantly. This pattern of the increased gap is similar to the findings discussed in [29], [31]. On the other hand, the proposed DGN continuously decreases the gap and further improves the segmentation performance. As can be seen, in comparison to the fine-tuning based and C&D based strategies, the proposed DGN is more effective in degraded image semantic segmentation. As shown in Fig. 3, we provide sample qualitative segmentation results for demonstration. In comparison to the segmentation results based on the network fine-tuning, the proposed DGN obtains visually better results, especially when the degradation degree is high, and it can accurately classify the components when using network fine-tuning and render more accurate segmentation results.

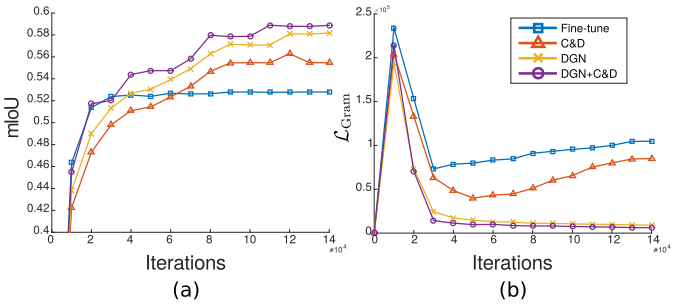


Fig. 4. (a) The mIoUs (in percentage) of segmenting degree-$\mathbf{d}_5$ Gaussian blur images sythesized from *PASCAL VOC 2012* dataset using: 1) Baseline networks fine-tuned using the degraded images; 2) Baseline networks trained using both clean and degraded images; 3) DGN trained using the degraded images; 4) DGN trained using paired clean and degraded images. (b) The dense-Gram loss $\mathcal{L}_{\text{Gram}}$ over 140,000 training iterations. The baseline segmentation network is FCN8s.

D. Impact of Image Restoration Based Pre-Processing

We conduct experiments to evaluate whether the image restoration based pre-processing could help the degraded image segmentation. Note that we select FCN8s as the baseline network for validation because it is fast to train and is well-studied by the community. The Gaussian blurred images are deblurred using the conventional deconvbind,[4] linear motion blurred images are deblurred using DeblurGAN [44], images with salt & pepper noise are resorted using median filter,[5] and hazy images are dehazed using: CAP dehaze [45], DehazeNet [46], and DCPDN [47]. The experiments are conducted in three respects: 1) Test the restored images using the model trained on the clean images; 2) Fine-tune the network using the restored images; 3) Train the proposed DGN using the restored images.

The quantitative results are reported in Table III. It is not surprising to observe a relative poor segmentation performance when directly test the restored images using the model pre-trained on the clean image. This is simply because that the

[4]https://www.mathworks.com/help/images/ref/deconvblind.html
[5]https://www.mathworks.com/help/images/ref/medfilt2.html

TABLE III

THE MIOUS (IN PERCENTAGE) OF SEGMENTING DEGRADED IMAGES USING: 1) PRE-TRAINED MODEL TESTED USING THE RESTORED IMAGES; 2) BASELINE NETWORK FINE-TUNED USING THE RESTORED IMAGES (*+FINE-TUNE); 3) DGN TRAINED USING THE DEGRADED IMAGES (DGN); 4) DGN TRAINED USING THE RESTORED IMAGES (*+DGN). THE FIVE DEGRADATION DEGREES ARE DENOTED USING $\mathbf{d}_1$, $\mathbf{d}_2$, $\mathbf{d}_3$, $\mathbf{d}_4$, AND $\mathbf{d}_5$, RESPECTIVELY. THE NUMBERS WITH BETTER AND THE BEST PERFORMANCE ARE HIGHLIGHTED IN BLUE AND RED, RESPECTIVELY

		PASCAL VOC 2012					SUNRGBD					CamVid					CityScapes				
		$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$
Gaussian Blur	Deconvbind	56.0	38.6	28.6	19.5	12.6	22.1	18.9	15.9	12.5	9.2	49.0	39.6	34.2	29.4	27.3	60.1	51.9	48.6	45.1	43.4
	Deconvbind+fine-tune	67.0	63.8	60.4	58.8	55.7	23.2	21.1	20.3	20.1	19.7	53.1	49.2	47.3	45.8	43.7	65.2	64.4	63.2	63.1	62.5
	DGN	**67.1**	**66.2**	**64.7**	**61.3**	**57.4**	**24.9**	**24.1**	**23.2**	**22.1**	**21.3**	**54.2**	**51.9**	**49.5**	**48.2**	**47.5**	**65.2**	**64.5**	**64.2**	**63.8**	**63.2**
	Deconvbind+DGN	**67.1**	**66.3**	**64.6**	**61.5**	**57.5**	**24.8**	**24.1**	**23.3**	**21.7**	**21.2**	**54.1**	**52.3**	**50.0**	**48.4**	**47.9**	**65.3**	**64.6**	**64.3**	**63.9**	**63.4**
Motion Blur	DeBlurGAN	60.9	45.4	33.3	24.4	16.5	23.9	20.9	18.5	15.9	12.9	53.2	51.1	46.6	43.2	42.3	63.3	63.0	62.9	60.3	60.8
	DeBlurGAN+fine-tune	65.2	58.8	53.9	51.4	46.2	**24.7**	20.8	19.2	18.3	17.1	54.2	**52.2**	47.1	44.7	42.9	65.1	63.8	63.1	62.8	61.9
	DGN	**65.4**	**59.7**	**54.7**	**52.2**	**48.1**	23.6	**22.2**	**20.1**	**19.1**	**17.8**	**54.2**	48.9	**47.2**	**45.1**	**43.7**	**65.2**	**64.5**	**63.7**	**63.1**	**62.3**
	DeBlurGAN+DGN	**65.9**	**60.3**	**54.2**	**51.6**	**48.3**	**25.7**	**25.1**	**23.2**	**19.4**	**17.2**	**55.3**	**53.9**	**50.3**	**45.6**	**43.5**	**65.4**	**64.7**	**64.1**	**63.3**	**62.6**
S & P Noise	Median Filter	60.6	56.0	52.1	47.6	43.0	23.9	22.4	21.3	20.0	18.6	52.6	52.4	52.3	52.3	51.9	65.1	65.0	64.6	62.3	62.0
	Median Filter+fine-tune	66.6	66.1	65.2	64.3	62.4	**24.1**	**22.9**	**21.8**	20.1	19.2	**53.1**	**52.9**	**52.7**	**52.6**	**52.2**	**65.2**	**65.2**	**64.7**	**63.2**	**62.6**
	DGN	**66.9**	**66.4**	**66.0**	**65.1**	**64.2**	21.9	21.7	21.1	**20.3**	**20.1**	51.9	50.7	50.2	50.1	49.6	64.9	63.2	62.7	60.1	59.7
	Median FIlter+DGN	**67.1**	**66.9**	**66.7**	**65.7**	**65.1**	**24.9**	**24.2**	**22.4**	**21.1**	**20.7**	**56.3**	**56.1**	**56.2**	**55.7**	**55.2**	**65.3**	**65.3**	**64.9**	**63.3**	**62.7**
Haze	CAP Dehaze	64.6	62.4	58.9	53.6	47.9	23.5	22.8	21.8	20.7	19.4	46.2	40.8	35.7	30.6	25.8	56.7	51.3	46.6	41.0	35.8
	DehazeNet	66.3	63.3	59.2	54.1	48.8	23.9	23.1	22.1	21.0	19.7	47.8	43.3	38.7	34.0	29.3	56.8	51.7	46.5	41.5	36.1
	DCPDN	66.5	65.3	61.3	57.2	52.4	24.1	23.3	22.2	21.4	20.2	50.7	49.8	43.2	33.7	30.2	61.5	55.5	50.9	45.5	40.7
	CAP Dehaze+fine-tune	66.5	65.2	64.1	61.7	61.1	25.7	23.2	22.3	21.9	20.8	53.7	52.9	51.9	50.0	49.1	64.8	63.2	62.1	60.9	59.2
	DehazeNet+fine-tune	66.7	65.5	64.4	62.1	61.9	25.8	23.5	23.1	22.3	21.4	53.8	53.1	52.3	50.9	50.0	64.5	63.4	62.8	61.9	60.2
	DCPDN+fine-tune	67.0	65.6	64.8	62.7	62.3	25.8	23.1	23.3	22.3	21.6	53.8	53.2	52.1	50.1	50.2	65.0	63.5	63.1	62.2	61.1
	DGN	66.7	65.7	64.9	63.3	63.1	25.8	23.7	23.5	22.8	22.3	53.8	53.2	**52.5**	**51.8**	**51.2**	64.2	63.8	**63.2**	62.3	61.4
	CAP Dehaze+DGN	66.9	66.2	65.0	63.2	63.2	26.3	24.4	23.6	22.9	22.0	53.9	53.3	52.4	51.7	50.5	**65.0**	63.6	62.7	62.5	61.6
	DehazeNet+DGN	**67.0**	**66.6**	**65.6**	**64.4**	**63.6**	**26.8**	**25.2**	**23.8**	**23.0**	**22.4**	**54.2**	**54.0**	52.2	51.6	51.1	64.9	**64.0**	63.1	**62.7**	**61.8**
	DCPDN+DGN	**67.2**	**66.9**	**65.4**	**64.1**	**63.7**	**27.1**	**26.8**	**24.2**	**23.1**	**22.4**	**54.9**	**54.1**	**53.3**	**51.9**	**51.5**	**65.2**	**64.7**	**63.8**	**63.4**	**62.4**

image restoration based pre-processing usually cannot completely restore the degraded images to their clean counterparts. Not to mention that the image restoration based pre-processing can potentially modify both texture and color information of the image and could introduce additional noise to the restored images, which result in a relative poor segmentation performance.

One exception is observed when using the median filter to remove the salt & pepper noise on CamVid and CityScapes datasets. The segmentation performance using restored images outperforms the proposed DGN. This is because that median filter is very effective in removing salt & pepper noise. Qualitatively, the restored images and the original images are visually identical. Quantitatively, in comparison to the Structure Similarity (SSIM) [48] of the other degradation effects (averagely SSIM $=$ 0.591), the SSIM between the restored image using the median filter and the clean images is 0.863, where SSIM $=$ 1 indicates that two images are completely identical. Therefore, we can expect the differences between the restored images and the clean image is small, and the segmentation performance of the restored images is high.

Fine-tuning the network using the restored image improves the performance by a large margin. However, the remaining differences between the restored images and the clean images pose an obstacle in improving the performance. To justify this point of view, we train the proposed DGN using the restored images. If there exists no or little gap in the distributions of the features learned using the restored images and the clean images, we can expect the risks of the source and target network to be very similar to each other, such that $\delta_{\text{Gram}} \approx 0$. This indicates that, in comparison to the results fine-tuned using the restored images, we are expecting little or no improvement when training the proposed DGN using the restored images. However, as shown in Table III, the proposed DGN trained on the restored images constantly achieves the best performance and outperforms the approaches fine-tuned on the restored images by averagely 1.9%, 1.0%, 1.9%, and 1.2% for the *PASCAL VOC 2012*, *SUNRGBD*, *CamVid*, and *CityScapes* datasets, respectively. Therefore, we conclude that the differences between the clean and restored images still hinder the performance from further improvement. The proposed DGN is demonstrated to be effective in degraded image semantic segmentation and can further improve the degraded image semantic segmentation performance when using the restored images.

E. Impact of Gram Matrix & Dense-Interweaving Matching

To validate the impact of the Gram matrix, we conduct the experiments in two respects. Firstly, we train the network (DGN-MSE) by directly minimizing the Mean Square Error (MSE) between the source and target feature maps, without using the Gram matrices, in the dense-interweaving manner. As shown in Table IV, when the degradation degree is $\mathbf{d}_5$, in comparison to DGN-MSE, the proposed DGN that uses the Gram matrix improves the segmentation performance by averagely 2.7%.

Secondly, as discussed in [39], matching the Gram matrices can be considered as a maximum mean discrepancy process [49] with the second order polynomial kernel. Similar to [39], we conduct the experiments by adopting 1) linear kernel (DGN-Linear) and 2) Gaussian kernel (DGN-Gaussian) for evaluation. Note that, as the Gram matrix based maximum mean discrepancy and MSE are different in both definition and calculation, the DGN-MSE and DGN-Linear are also different. Quantitatively, as shown in Table IV, we achieve comparable segmentation performance when using the different kernels. We conclude that, in the proposed DGN, using either the polynomial (default), the linear, or the Gaussian kernels does *not* lead to significant changes in segmentation performance.

To validate the impact of the dense-interweaving matching, we modify the proposed DGN and train the network by layer-wisely matching the Gram matrices (DGN-Layerwise),

TABLE IV

THE MIOUS (IN PERCENTAGE) OF SEGMENTING DEGRADED IMAGES USING: 1) DGN; 2) DIRECT FEATURE MAPS MATCHING (DGN-MSE); 3) LINEAR KERNEL (DGN-LINEAR); 4) GAUSSIAN KERNELS (DGN-GAUSSIAN); 5) LAYER-WISE GRAM MATRICES MATCHING (DGN-LAYERWISE). FOR EACH DEGRADATION EFFECT, THE DEGRADATION DEGREE IS INCREASED FROM LEFT TO RIGHT. THE NUMBERS WITH THE BEST PERFORMANCE ARE HIGHLIGHTED IN RED

		PASCAL VOC 2012					SUNRGBD					CamVid					CityScapes				
		$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$
Gaussian Blur	DGN	**67.5**	**66.6**	64.3	**61.5**	57.3	25.0	**24.2**	23.5	22.0	20.8	54.3	**52.2**	**49.6**	47.7	46.6	65.2	64.5	**64.2**	63.8	**63.2**
	DGN-MSE	67.0	64.8	61.4	53.7	51.7	23.4	22.1	20.7	19.1	17.8	53.2	51.0	48.2	45.2	44.3	63.7	63.6	62.9	60.8	59.6
	DGN-Linear	67.1	66.2	**64.7**	61.3	57.4	24.9	24.1	23.2	22.1	**21.0**	54.2	51.9	49.5	**48.2**	47.5	**65.3**	**64.6**	64.1	**63.9**	63.1
	DGN-Gaussian	67.0	66.4	64.6	61.2	**57.7**	**25.0**	23.9	**23.6**	**22.2**	20.5	**54.6**	51.6	49.5	48.0	**47.9**	65.0	64.5	64.2	63.7	63.0
	DGN-Layerwise	67.0	65.9	64.2	60.9	57.1	24.6	23.7	22.6	21.4	20.1	53.9	51.6	49.1	47.9	47.1	64.6	63.9	63.7	63.3	62.3
Motion Blur	DGN	65.3	59.4	54.4	**52.6**	48.0	23.2	21.7	**20.2**	19.1	17.2	**54.6**	48.5	47.1	45.0	43.7	65.2	64.5	63.7	63.1	**62.3**
	DGN-MSE	64.5	58.2	52.3	49.6	44.5	22.4	20.8	19.6	17.8	16.9	52.8	46.0	45.2	43.1	42.0	64.1	61.8	61.8	60.9	60.4
	DGN-Linear	65.4	59.7	**54.7**	52.2	**48.1**	23.6	**22.2**	20.1	**19.1**	17.8	54.2	**48.9**	47.2	45.1	**43.7**	**65.3**	64.3	**63.8**	**62.9**	62.2
	DGN-Gaussian	**65.5**	**59.9**	54.6	52.1	47.8	**23.7**	22.1	20.0	19.0	**18.3**	54.1	48.7	**47.3**	**45.5**	43.5	65.0	**64.6**	63.7	63.2	62.3
	DGN-Layerwise	65.4	59.2	54.3	51.3	47.5	23.4	21.7	19.5	18.4	17.2	53.8	48.5	46.9	44.7	43.5	65.0	64.5	63.8	62.6	62.2
S & P Noise	DGN	66.5	**66.7**	66.0	**65.4**	64.1	21.8	21.6	20.6	20.0	**20.5**	51.9	50.9	50.1	**50.8**	49.3	**64.9**	**63.2**	62.7	**60.1**	59.7
	DGN-MSE	66.5	64.9	64.7	62.9	61.1	21.5	21.0	20.3	19.0	18.7	51.7	50.3	48.0	46.7	46.1	64.4	63.1	60.3	56.4	56.6
	DGN-Linear	**66.9**	66.4	**66.0**	65.1	64.2	**21.9**	**21.7**	21.1	**20.3**	20.1	**51.9**	50.7	**50.2**	50.1	**49.6**	64.6	62.7	62.6	59.9	59.6
	DGN-Gaussian	66.7	66.5	65.7	65.3	**64.2**	21.6	21.3	**21.3**	19.9	20.3	51.9	**51.2**	49.8	50.0	49.5	64.8	63.0	**62.9**	59.6	**59.7**
	DGN-Layerwise	66.5	65.8	65.1	64.6	63.9	21.3	21.4	20.8	19.6	19.2	51.7	50.5	49.8	49.7	49.3	64.3	62.9	62.3	59.9	59.0
Haze	DGN	66.3	66.0	**65.1**	62.9	**63.5**	25.6	23.9	**23.7**	**22.9**	22.1	**54.0**	53.0	**53.0**	**52.3**	51.2	64.2	63.8	**63.2**	62.3	61.4
	DGN-MSE	65.3	63.9	62.2	61.4	60.7	25.7	22.9	22.5	20.9	20.3	53.7	52.1	51.2	50.5	49.5	63.6	62.5	62.3	60.9	60.0
	DGN-Linear	**66.7**	65.7	64.9	**63.3**	63.1	**25.8**	23.7	23.5	22.8	22.3	53.8	53.2	52.5	51.7	**51.2**	64.1	**63.9**	63.1	62.2	61.4
	DGN-Gaussian	66.1	**66.2**	64.3	63.2	62.4	25.1	**23.9**	23.4	22.9	**22.6**	53.4	**53.5**	51.9	52.0	51.1	**64.2**	63.6	63.0	**62.4**	**61.5**
	DGN-Layerwise	65.9	65.1	64.2	62.6	62.5	25.4	23.3	23.1	22.1	21.6	53.2	52.6	51.9	51.1	50.3	63.5	63.3	62.3	61.3	60.7

TABLE V

THE MIOUS (IN PERCENTAGE) OF SEGMENTING DEGRADED IMAGES USING DIFFERENT λ IN EQ. (5), DEFAULT $\lambda = 1 \times 10^{-1}$. THE FIVE DEGRADATION DEGREES ARE DENOTED USING $\mathbf{d}_1$, $\mathbf{d}_2$, $\mathbf{d}_3$, $\mathbf{d}_4$, AND $\mathbf{d}_5$, RESPECTIVELY. THE NUMBERS WITH THE BEST PERFORMANCE ARE HIGHLIGHTED IN RED

		PASCAL VOC 2012					SUNRGBD					CamVid					CityScapes				
		$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$
Gaussian Blur	$\lambda = 10^{-3}$	67.0	64.2	60.8	56.4	53.1	23.8	22.1	20.6	19.6	18.3	53.1	49.1	46.3	43.8	41.8	63.7	61.5	61.0	59.2	57.5
	$\lambda = 10^{-2}$	67.1	64.6	62.1	57.3	54.2	24.0	22.5	21.2	20.2	18.9	53.3	49.8	47.0	44.8	43.1	64.0	62.1	61.3	60.2	58.5
	$\lambda = 10^{-1}$	**67.1**	**66.2**	64.7	**61.3**	**57.4**	**24.9**	**24.1**	**23.2**	22.1	**21.0**	**54.2**	**51.9**	**49.5**	48.2	**47.5**	**65.2**	**64.5**	**64.2**	**63.8**	**63.2**
	$\lambda = 0.5$	67.1	66.1	**64.9**	61.3	57.3	24.7	24.1	23.2	**22.3**	21.0	54.0	51.7	49.4	48.2	47.5	65.0	64.4	64.2	63.7	63.1
	$\lambda = 1$	67.0	66.1	64.8	61.2	57.4	24.8	24.0	23.1	22.2	20.8	54.1	51.8	49.4	**48.3**	47.5	64.6	64.4	63.7	63.7	62.8
	$\lambda = 10$	67.0	64.5	61.6	57.2	53.9	23.7	22.3	21.2	20.0	18.5	53.3	49.7	46.8	44.7	42.7	63.9	62.3	61.1	60.1	58.9
Motion Blur	$\lambda = 10^{-3}$	64.9	58.5	52.2	49.9	45.4	23.1	20.8	18.9	17.5	16.2	51.6	46.2	45.2	43.7	41.5	62.4	61.4	61.9	61.4	60.3
	$\lambda = 10^{-2}$	65.0	58.8	52.8	50.5	46.1	23.3	21.2	19.2	17.9	16.6	52.3	46.9	45.8	44.1	42.1	63.0	62.6	61.8	62.0	60.4
	$\lambda = 10^{-1}$	**65.4**	**59.7**	**54.7**	**52.2**	**48.1**	**23.6**	**22.2**	**20.1**	19.1	17.8	**54.2**	**48.9**	**47.2**	**45.1**	43.7	**65.2**	**64.5**	63.7	63.1	62.3
	$\lambda = 0.5$	65.3	59.6	54.6	52.1	48.0	23.4	22.2	20.0	**19.2**	17.8	54.2	48.9	47.1	44.9	**43.7**	65.1	64.4	63.8	63.2	62.5
	$\lambda = 1$	65.3	59.5	54.6	52.0	47.9	23.5	22.0	20.0	18.9	**17.9**	54.1	48.8	47.1	45.0	43.7	64.8	64.3	**63.9**	**63.3**	**62.6**
	$\lambda = 10$	64.8	58.3	52.2	50.2	45.9	23.0	21.0	19.1	17.4	16.5	52.1	46.3	45.7	43.5	41.9	63.0	61.5	61.8	61.0	60.9
S & P Noise	$\lambda = 10^{-3}$	66.3	65.5	64.6	63.4	61.7	21.0	19.9	19.8	18.7	19.0	51.8	50.0	48.4	47.4	46.3	64.5	62.7	61.3	57.2	55.9
	$\lambda = 10^{-2}$	66.5	65.8	65.0	63.8	62.3	21.2	20.4	20.1	19.1	19.3	51.8	50.2	48.9	48.1	47.1	**65.2**	62.2	61.5	58.0	57.4
	$\lambda = 10^{-1}$	**66.9**	**66.4**	**66.0**	65.1	64.2	**21.9**	**21.7**	**21.1**	20.3	**20.1**	**51.9**	50.7	**50.2**	**50.1**	**49.6**	64.9	63.2	**62.7**	**60.1**	**59.7**
	$\lambda = 0.5$	66.8	66.3	66.0	**65.3**	64.0	21.7	21.5	21.0	**20.7**	20.1	51.7	50.7	50.1	50.1	49.5	64.8	63.1	62.6	60.1	59.6
	$\lambda = 1$	66.8	66.3	65.8	65.2	**64.2**	21.8	21.6	21.0	20.5	19.9	51.8	**50.8**	50.1	50.0	49.5	64.5	**63.5**	62.5	59.7	59.1
	$\lambda = 10$	66.5	65.7	64.8	63.4	61.8	21.1	20.0	20.0	18.8	19.1	51.8	50.1	48.6	47.4	46.4	65.3	62.6	60.7	57.0	56.2
Haze	$\lambda = 10^{-3}$	65.6	63.9	62.2	60.5	60.4	25.5	22.5	22.2	21.5	20.4	53.6	52.3	50.7	49.6	48.9	**64.4**	62.7	61.6	60.2	59.3
	$\lambda = 10^{-2}$	65.9	64.4	62.9	61.2	61.1	25.6	22.8	22.5	21.8	20.9	53.7	52.6	51.1	50.1	49.5	64.4	63.1	61.6	60.4	59.7
	$\lambda = 10^{-1}$	**66.7**	**65.7**	**64.9**	63.3	63.1	**25.8**	23.7	23.5	22.8	22.3	**53.8**	**53.2**	**52.5**	**51.7**	51.2	64.2	**63.8**	**63.2**	62.3	61.4
	$\lambda = 0.5$	66.6	65.5	64.7	**63.5**	63.1	25.6	**23.8**	23.4	**22.9**	22.3	53.8	53.1	52.5	51.6	**51.5**	64.1	63.7	63.1	**62.6**	61.2
	$\lambda = 1$	66.6	65.6	64.8	63.2	**63.2**	25.7	23.6	**23.6**	22.7	**23.4**	53.7	53.1	52.4	51.6	51.3	63.9	63.6	62.7	62.5	**61.6**
	$\lambda = 10$	65.9	64.2	62.7	61.2	60.8	25.5	22.6	22.2	21.5	20.7	53.5	52.3	51.1	50.1	49.4	63.5	63.0	62.2	61.1	60.0

i.e., the Gram matrix of the feature maps of one layer in the target network is matched to its corresponding Gram matrix of the same layer in the source network. In comparison to DGN-Layerwise, the proposed DGN which involves the dense-interweaving matching improves the segmentation performance by averagely 0.6%.

F. Impact of Hyperparameter λ Selection

We conduct experiments to evaluate the impact of hyperparameter λ by using different values $\lambda \in \{10^{-3}, 10^{-2}, 10^{-1}, 0.5, 1, 10\}$ in Eq. (5). As shown in Table V, by increasing λ, the performance first increases. However, further increasing λ would force the network to put more efforts on minimizing the dense-Gram loss, which results in the decrease of the power of optimizing the target network in semantic segmentation task. Based on the results shown in Table V, we select $\lambda = 10^{-1}$ as default. For the other baseline networks, we also use the same $\lambda = 10^{-1}$ for all the experiments.

G. Impact of Dense-Gram Block Selection

We conduct additional experiments to evaluate the segmentation performance when the dense-Gram matching starts at the different convolutional blocks. As the dense-Gram matching tends to force the feature distribution in the target network to be similar to the feature distribution in the source network, the dense-Gram matching can be considered as a form of regularization. Specifically, when the dense-Gram matching starts at a higher block, the proposed DGN allows the target network to learn the features with more freedom, and vice versa.

The quantitative results are shown in Table VI. Let "DGN-B2", "DGN-B3", "DGN-B4" (default), "DGN-B5", "DGN-B6" denote the proposed DGN with the dense-Gram matching starting at the 2^{nd}, 3^{rd}, 4^{th}, 5^{th}, and

TABLE VI

THE MIOUS (IN PERCENTAGE) OF SEGMENTING DEGRADED IMAGES WHEN THE DENSE-GRAM MATCHING BEGINS AT THE 2^{nd}, 3^{rd}, 4^{th}, 5^{th}, AND 6^{th} CONVOLUTIONAL BLOCKS. FOR EACH DEGRADATION EFFECT, THE DEGRADATION DEGREE IS INCREASED FROM LEFT TO RIGHT. THE NUMBERS WITH THE BEST PERFORMANCE ARE HIGHLIGHTED IN RED

		PASCAL VOC 2012					SUNRGBD					CamVid					CityScapes				
		$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$
Gaussian Blur	DGN-B2	**67.1**	**66.8**	62.7	57.6	55.2	**25.1**	24.1	21.2	20.3	19.7	54.5	52.0	48.1	44.9	44.4	**65.3**	**64.9**	62.7	60.4	59.9
	DGN-B3	67.1	66.6	63.2	58.3	55.9	25.0	**24.1**	22.4	21.7	20.3	**54.5**	**52.1**	48.9	45.1	45.8	65.3	64.8	63.5	60.5	61.2
	DGN-B4	67.1	66.2	**64.7**	**61.3**	**57.4**	24.9	24.1	**23.2**	**22.1**	21.0	54.2	51.9	49.5	**48.2**	**47.5**	65.2	64.5	**64.2**	**63.8**	63.2
	DGN-B5	67.1	65.8	63.8	60.8	57.3	24.8	24.0	23.2	22.0	**21.2**	54.2	51.8	**49.7**	48.0	47.5	65.1	64.6	64.2	63.3	**63.3**
	DGN-B6	67.0	64.2	62.5	59.1	56.2	24.4	23.7	22.7	21.8	20.8	54.1	50.7	49.2	47.5	44.7	65.1	63.1	63.7	62.8	60.1
Motion Blur	DGN-B2	65.4	59.8	54.4	50.1	44.2	**23.6**	22.3	19.3	18.1	16.9	**54.4**	**49.3**	46.3	44.1	42.2	**65.7**	64.5	63.1	62.5	60.5
	DGN-B3	**65.5**	**59.8**	54.6	50.7	45.3	23.6	**22.3**	19.7	18.3	17.3	54.3	49.0	47.0	44.5	43.1	65.6	**64.7**	63.7	62.6	61.6
	DGN-B4	65.4	59.7	54.7	**52.2**	**48.1**	23.6	22.2	**20.1**	**19.1**	**17.8**	54.2	48.9	**47.2**	**45.1**	**43.7**	65.2	64.5	**63.7**	**63.1**	62.3
	DGN-B5	65.3	59.2	**54.9**	52.1	47.9	23.6	22.1	20.0	19.0	17.6	54.1	49.0	47.0	44.8	43.2	65.5	64.1	63.6	62.7	**62.4**
	DGN-B6	64.9	58.9	54.5	51.4	46.8	23.3	21.9	19.4	18.7	17.2	53.7	48.7	47.1	43.5	42.7	64.7	64.7	64.0	62.0	61.2
S & P Noise	DGN-B2	66.9	**66.7**	64.6	63.3	62.3	**22.1**	**21.9**	20.1	19.1	19.3	**52.1**	**51.1**	48.9	48.7	47.5	64.7	63.6	61.5	58.6	57.4
	DGN-B3	**66.9**	66.6	65.0	64.2	63.8	21.9	21.6	20.8	19.7	19.6	52.0	50.9	49.6	49.2	48.8	64.8	63.1	61.9	59.2	58.9
	DGN-B4	66.9	66.4	**66.0**	65.1	64.2	21.9	21.7	**21.1**	20.3	**20.1**	51.9	50.7	**50.2**	**50.1**	**49.6**	**64.9**	**63.2**	**62.7**	60.1	59.7
	DGN-B5	66.9	66.4	65.8	**65.4**	**64.5**	21.9	21.3	21.1	**20.5**	20.0	51.8	50.7	49.9	50.0	49.1	64.3	62.8	62.6	**60.1**	**59.7**
	DGN-B6	66.8	66.1	64.8	65.3	63.1	21.4	20.6	20.6	19.8	19.8	51.8	50.2	49.6	49.3	49.0	64.4	62.9	62.4	59.0	58.6
Haze	DGN-B2	**66.9**	65.5	63.7	61.7	61.1	**25.9**	**23.8**	23.1	21.3	20.9	53.8	**53.3**	51.3	50.3	49.3	64.1	**64.3**	62.4	60.8	59.0
	DGN-B3	66.8	65.2	63.1	62.9	62.7	25.8	23.7	23.3	21.9	21.8	**54.1**	53.2	51.8	50.9	50.5	**64.3**	63.6	62.2	61.8	60.4
	DGN-B4	66.7	**65.7**	64.9	**63.3**	**63.1**	25.8	23.7	**23.5**	22.8	**22.3**	53.8	53.2	**52.5**	**51.7**	**51.2**	64.2	63.8	**63.2**	**62.3**	**61.4**
	DGN-B5	66.6	65.6	**65.0**	63.1	63.0	25.8	23.6	23.5	**22.6**	22.1	53.9	53.1	52.0	51.5	50.6	64.2	63.5	62.8	62.1	61.1
	DGN-B6	66.6	64.9	64.1	62.9	61.8	25.6	22.9	23.3	22.2	21.8	53.2	53.1	52.1	50.8	49.9	63.2	63.5	62.7	61.7	60.0

6^{th} convolutional blocks, respectively. When the degradation degree is small, the dense-Gram matching that starts at a lower block (e.g., DGN-B2) strengthens the ability of the feature regulation and improves the segmentation performance. On the other hand, when the degradation degree is high, the dense-Gram matching that starts at a higher block (e.g., DGN-B6) decreases the ability of feature regulation and decreases the segmentation performance. As shown in Table VI, we observe that the DGN-B4 (middle block) provides the best overall performance. For the other segmentation network, we start the dense-Gram matching at the block located in the middle of the network.

H. Impact of Learning Speed Tuning

Learning speed tuning can be considered as an alternative way of addressing the minimization of the gap in feature distributions of higher layers [30], [33]. Intuitively, to preserve the feature distribution in higher layers, one "naive" way is to manually tune down the learning rate of the higher layers, such that the weight updating speed in higher layers is slow. However, manually tuning the learning rate is heuristic and laborious. A "smart" way of tuning the learning speed is to selectively slow down the learning of network weights by using the Elastic Weight Consolidation (EWC) module [33], such that the network can remember the features learned using the clean images. However, the employment of the EWC module requires additional approximately three times as many parameters as the original network. This level of GPU memory consumption poses a potential obstacle in implementing the EWC modules.

Firstly, we conduct experiments to evaluate the impact of using the smaller learning rates during fine-tuning. For fair comparisons, we only tune down the learning rate of the layers that is associated with the dense-Gram matchings. The learning rate of those higher layers is reduced by multiplying a constant ratio. In this paper, we select the ratio to be $\rho \in \{1, 10^{-1}, 10^{-2}, 0\}$, where $\rho = 0$ denotes the weights of higher layers are fixed during fine-tuning and $\rho = 1$ denotes the network using the same learning rate for the whole network fine-tuning. Secondly, we conduct an experiment by employing the EWC module into the segmentation network. We follow the same experiment setting used in network fine-tuning, and select the weight of the EWC loss $\lambda_{\text{EWC}} = 400$ [33].

Quantitatively, as shown in Table VII, in comparison to the default network fine-tuning ($\rho = 1$), tuning down the learning rate of the higher blocks with $\rho = 10^{-1}$ improves the performance. However, further tuning down the learning rate (e.g., $\rho = 10^{-2}$ and $\rho = 0$) decreases the performance. In comparison to the fine-tuning based approaches, the employment of EWC module achieves the second best performance. All in all, when the degradation degree is $\mathbf{d}_5$, the proposed DGN outperforms the learn rate tuning based approaches by averagely 3.0%, and constantly outperforms the EWC based approach by averagely 1.7%.

I. Evaluation on Real Haze Images

To further demonstrate the effectiveness of the proposed DGN, we evaluate the porposed method using the 100 real haze images[6] mined from the Internet. Specifically, the mined real haze images are annotated following PASCAL VOC 2012 dataset criteria. For fair comparison, we do not apply the image-restoration processing to the real haze image dataset during the evaluation. Since it is difficult to quantify the degradation degree on the real haze images, we directly deploy five models, denoted by $\mathbf{d}_1$, $\mathbf{d}_2$, $\mathbf{d}_3$, $\mathbf{d}_4$, and $\mathbf{d}_5$, respectively, that were trained using the corresponding-degree hazy images synthesized from PASCAL VOC 2012 and report their testing performances on the 100 real images in Table VIII. We note that when degradation degree is $\mathbf{d}_3$, the proposed DGN constantly shows the best segmentation performance. We assume that the degradation degree of the real images are similar to the sythnesized degree-$\mathbf{d}_3$ haze images. As the real haze images are different from the synthesized haze images, we are expecting minor performance decreases.

[6]https://cvl.cse.sc.edu/Download/data_annotated_voc.tar.gz

TABLE VII

THE MIOUS (IN PERCENTAGE) OF SEGMENTING DEGRADED IMAGES USING DIFFERENT ρ. LET "EWC" DENOTE THE SEGMENTATION NETWORK THAT EMPLOYS THE EWC MODULE. THE FIVE DEGRADATION DEGREES ARE DENOTED USING $\mathbf{d}_1$, $\mathbf{d}_2$, $\mathbf{d}_3$, $\mathbf{d}_4$, AND $\mathbf{d}_5$, RESPECTIVELY. THE NUMBERS WITH BETTER AND THE BEST PERFORMANCE ARE HIGHLIGHTED IN BLUE AND RED, RESPECTIVELY

		PASCAL VOC 2012					SUNRGBD					CamVid					CityScapes				
		$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$
Gaussian Blur	$\rho=1$	67.0	64.1	60.7	56.4	52.8	23.7	21.9	20.5	19.6	18.2	53.0	49.1	46.2	43.7	41.7	64.4	61.9	61.3	59.5	57.6
	$\rho=10^{-1}$	67.1	65.0	60.8	56.2	52.7	23.8	22.2	20.5	19.7	18.3	54.1	51.1	48.1	45.0	42.6	**65.3**	63.7	62.8	61.0	58.2
	$\rho=10^{-2}$	67.1	64.7	60.0	53.9	51.2	24.2	22.5	20.4	18.9	17.8	54.1	51.0	47.8	44.7	42.3	65.1	64.0	62.4	60.3	57.9
	$\rho=0$	67.1	64.6	59.8	53.8	51.1	24.6	22.7	20.4	18.7	17.4	54.0	50.8	47.5	44.2	41.1	65.0	63.5	62.2	60.1	56.7
	EWC	**67.1**	**65.9**	**62.9**	**59.2**	**55.6**	**24.8**	**23.5**	**23.0**	**21.1**	**18.7**	**54.2**	**51.6**	**49.1**	**47.1**	**45.0**	65.2	**64.3**	**64.2**	**62.7**	**61.9**
	DGN	**67.1**	**66.2**	**64.7**	**61.3**	**57.4**	**24.9**	**24.1**	**23.2**	**22.1**	**21.0**	**54.2**	**51.9**	**49.5**	**48.2**	**47.5**	**65.4**	**64.5**	**64.2**	**63.8**	**63.2**
Motion Blur	$\rho=1$	64.8	58.4	52.1	49.8	45.4	23.0	20.8	18.8	17.5	16.1	51.6	46.2	45.2	43.7	41.5	63.0	62.0	62.0	61.8	60.5
	$\rho=10^{-1}$	65.1	59.3	52.2	50.6	46.2	23.4	21.6	18.9	17.4	16.0	53.9	48.7	45.5	43.4	41.1	65.2	64.6	62.4	61.6	59.8
	$\rho=10^{-2}$	65.1	59.2	51.3	48.4	43.5	23.4	21.5	18.6	17.1	15.9	53.8	48.3	45.2	43.2	40.8	65.2	64.1	61.9	61.5	59.4
	$\rho=0$	64.9	59.2	50.7	47.1	40.9	23.4	21.2	18.2	17.2	15.8	53.6	48.1	45.1	43.2	40.6	65.0	64.2	61.7	61.4	59.6
	EWC	**65.3**	**59.4**	**53.9**	**51.2**	**46.8**	**23.6**	**22.0**	**19.7**	**18.8**	**16.5**	**54.1**	**48.8**	**46.3**	**44.5**	**42.3**	**65.5**	**64.7**	**62.9**	**62.9**	**60.9**
	DGN	**65.4**	**59.7**	**54.7**	**52.2**	**48.1**	**23.6**	**22.2**	**20.1**	**19.1**	**17.8**	**54.2**	**48.9**	**47.2**	**45.1**	**43.7**	**65.2**	**64.5**	**63.7**	**63.1**	**62.3**
S & P Noise	$\rho=1$	66.3	65.5	64.6	63.4	61.6	21.0	19.9	19.8	18.7	19.0	51.7	50.0	48.4	47.3	46.2	64.9	62.6	61.0	57.7	56.6
	$\rho=10^{-1}$	**66.8**	66.3	65.2	63.4	61.3	21.6	21.4	20.8	19.0	18.6	51.7	50.1	48.6	47.2	46.0	**65.0**	**63.0**	61.1	57.3	56.3
	$\rho=10^{-2}$	66.8	66.2	65.0	62.1	60.3	21.5	21.3	20.7	18.7	18.0	51.7	50.0	47.9	46.9	45.9	65.0	62.4	60.5	56.9	56.4
	$\rho=0$	66.7	66.1	64.9	61.7	59.7	21.4	21.1	20.3	18.6	17.8	51.7	49.8	47.8	46.9	45.8	64.6	62.4	60.3	57.3	56.2
	EWC	66.8	**66.4**	**65.4**	**64.7**	**63.1**	**21.7**	**21.4**	**20.9**	**19.5**	**19.2**	**51.8**	**50.5**	**49.7**	**49.1**	**47.2**	64.8	62.9	**62.2**	**59.4**	**57.5**
	DGN	**66.9**	**66.4**	**66.0**	**65.1**	**64.2**	**21.9**	**21.7**	**21.1**	**20.3**	**20.1**	**51.9**	**50.7**	**50.2**	**50.1**	**49.6**	**64.9**	**63.2**	**62.7**	**60.1**	**59.7**
Haze	$\rho=1$	65.6	63.9	62.2	60.5	60.4	25.5	22.5	22.2	21.5	20.4	53.6	52.3	50.7	49.6	48.9	63.9	62.9	61.5	60.6	59.1
	$\rho=10^{-1}$	66.4	65.0	62.7	60.8	60.2	25.7	23.6	22.4	21.7	20.6	53.8	53.0	51.4	49.6	48.6	64.2	63.0	62.0	60.2	59.0
	$\rho=10^{-2}$	66.3	64.9	62.5	60.4	60.0	25.7	23.5	22.1	21.5	20.0	53.8	52.9	51.2	49.4	48.7	64.2	63.4	62.3	60.3	59.3
	$\rho=0$	66.2	64.8	62.4	60.1	59.8	25.7	23.4	21.8	21.1	19.9	53.7	52.8	51.1	49.4	48.4	64.0	63.1	61.8	60.4	59.0
	EWC	**66.6**	**65.3**	**63.5**	**61.8**	**61.1**	**25.9**	**23.7**	**23.1**	**22.2**	**21.1**	**54.0**	**53.2**	**52.1**	**50.8**	**49.2**	**64.9**	**63.6**	**62.9**	**61.7**	**59.3**
	DGN	**66.7**	**65.7**	**64.9**	**63.3**	**63.1**	**25.8**	**23.7**	**23.5**	**22.8**	**22.3**	**53.8**	**53.2**	**52.5**	**51.7**	**51.2**	**64.2**	**63.8**	**63.2**	**62.3**	**61.4**

TABLE VIII

THE MIOUS (IN PERCENTAGE) OF SEGMENTING REAL HAZE IMAGES. $\mathbf{d}_1$, $\mathbf{d}_2$, $\mathbf{d}_3$, $\mathbf{d}_4$, AND $\mathbf{d}_5$ INDICATE THE FIVE DGN MODELS TRAINED USING THE CORRESPONDING-DEGREE HAZY IMAGES SYNTHESIZED FROM PASCAL VOC 2012. WITHOUT QUANTIFYING THE DEGRADATION DEGREES OF THE REAL IMAGES, WE DIRECTLY DEPLOY THE TRAINED MODELS FOR EVALUATION. THE NUMBERS WITH THE BEST PERFORMANCE ARE HIGHLIGHTED IN RED

	$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_3$	$\mathbf{d}_4$	$\mathbf{d}_5$
FCN8s+fine-tune	56.7	58.3	57.6	55.4	52.3
FCN8s+C&D	56.9	58.6	58.3	56.2	52.7
FCN8s+DGN	57.2	**58.8**	59.6	57.4	55.3
FCN8s+DGN+C&D	**57.5**	58.7	**60.2**	**59.2**	**56.4**
DeepLab v2+fine-tune	68.3	68.4	68.0	65.8	62.3
DeepLab v2+C&D	68.6	69.5	68.8	66.8	62.0
DeepLab v2+DGN	69.2	**69.6**	70.2	68.2	65.5
DeepLab v2+DGN+C&D	**69.5**	69.4	**70.4**	**69.7**	**66.1**
RefineNet+fine-tune	71.1	71.8	70.2	68.8	65.5
RefineNet+C&D	72.0	73.1	71.7	69.7	65.6
RefineNet+DGN	71.9	73.1	72.5	71.1	68.7
RefineNet+DGN+C&D	**72.6**	**73.2**	**73.2**	**72.6**	**69.4**
EncNet+fine-tune	73.2	73.8	74.0	71.3	67.6
EncNet+C&D	72.6	73.9	73.1	71.1	67.0
EncNet+DGN	72.8	74.0	74.2	72.6	70.0
EncNet+DGN+C&D	**72.9**	**74.0**	**75.0**	**74.1**	**71.2**
DeepLab v3 Plus+fine-tune	77.6	77.5	76.9	74.8	71.5
DeepLab v3 Plus+C&D	78.3	78.7	78.1	76.1	71.5
DeepLab v3 Plus+DGN	78.7	78.9	79.4	77.6	75.1
DeepLab v3 Plus+DGN+C&D	**78.8**	**79.0**	**80.1**	**79.7**	**75.5**

V. CONCLUSION

In this paper, we systematically study the problem of degraded image semantic segmentation and propose a Dense-Gram network to segment degraded images without using any image restoration based pre-processing when only the degraded images are available. The proposed DGN is evaluated using synthetic degraded images based on PASCAL VOC 2012, SUNRGBD, CamVid, and CityScapes benchmark datasets. In comparison to the network fine-tuning based, C&D based, image restoration based, and learning rate tuning based strategies, the proposed DGN substantially improves the semantic segmentation performance of the degraded images.

REFERENCES

[1] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 3431–3440.

[2] D. Guo *et al.*, "Automated lesion detection on MRI scans using combined unsupervised and supervised methods," *BMC Med. Imag.*, vol. 15, no. 1, p. 50, Dec. 2015.

[3] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, and J. Garcia-Rodriguez, "A review on deep learning techniques applied to semantic segmentation," 2017, *arXiv:1704.06857*. [Online]. Available: https://arxiv.org/abs/1704.06857

[4] C. Sakaridis, D. Dai, and L. Van Gool, "Semantic foggy scene understanding with synthetic data," *Int. J. Comput. Vis.*, vol. 126, no. 9, pp. 973–992, Sep. 2018.

[5] 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," 2016, *arXiv:1606.00915*. [Online]. Available: https://arxiv.org/abs/1606.00915

[6] G. Lin, A. Milan, C. Shen, and I. D. Reid, "RefineNet: Multi-path refinement networks for high-resolution semantic segmentation," in *Proc. CVPR*, Jul. 2017, pp. 1925–1934.

[7] D. Guo, K. Zheng, and S. Wang, "Lesion detection using T1-weighted MRI: A new approach based on functional cortical ROIs," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, pp. 4427–4431.

[8] D. Guo, L. Zhu, Y. Lu, H. Yu, and S. Wang, "Small object sensitive segmentation of urban street scene with spatial adjacency between object classes," *IEEE Trans. Image Process.*, vol. 28, no. 6, pp. 2643–2653, Jun. 2019.

[9] I. Vasiljevic, A. Chakrabarti, and G. Shakhnarovich, "Examining the impact of blur on recognition by convolutional networks," 2016, *arXiv:1611.05760*. [Online]. Available: https://arxiv.org/abs/1611.05760

[10] W. H. Richardson, "Bayesian-based iterative method of image restoration," *J. Opt. Soc. Amer.*, vol. 62, no. 1, pp. 55–59, Jan. 1972.

[11] D. Lu and Q. Weng, "A survey of image classification methods and techniques for improving classification performance," *Int. J. Remote Sens.*, vol. 28, no. 5, pp. 823–870, 2007.

[12] M. Roushdy, "Comparative study of edge detection algorithms applying on the grayscale noisy image using morphological filter," *GVIP J.*, vol. 6, no. 4, pp. 17–23, Dec. 2006.

[13] G. Celeux, F. Forbes, and N. Peyrard, "EM procedures using mean field-like approximations for Markov model-based image segmentation," *Pattern Recognit.*, vol. 36, no. 1, pp. 131–144, Jan. 2003.

[14] Y. Pei, Y. Huang, Q. Zou, Y. Lu, and S. Wang, "Does haze removal help CNN-based image classification," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Munich, Germany: Springer, Sep. 2018, pp. 682–697.

[15] 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.

[16] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," in *Proc. Int. Conf. Artif. Neural Netw.* Espoo, Finland: Springer, 2011, pp. 52–59.

[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[18] N. Damera-Venkata, T. D. Kite, W. S. Geisler, B. L. Evans, and A. C. Bovik, "Image quality assessment based on a degradation model," *IEEE Trans. Image Process.*, vol. 9, no. 4, pp. 636–650, Apr. 2000.

[19] L. Gatys, A. S. Ecker, and M. Bethge, "Texture synthesis using convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 262–270.

[20] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Proc. Eur. Conf. Comput. Vis.* Amsterdam, The Netherlands: Springer, 2016, pp. 694–711.

[21] L. A. Gatys, A. S. Ecker, M. Bethge, A. Hertzmann, and E. Shechtman, "Controlling perceptual factors in neural style transfer," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3985–3993.

[22] F. Yang, W. Choi, and Y. Lin, "Exploit all the layers: Fast and accurate CNN object detector with scale dependent pooling and cascaded rejection classifiers," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2129–2137.

[23] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. *The Pascal Visual Object Classes Challenge 2012 (VOC2012) Results*. Accessed: Sep. 3, 2012. [Online]. Available: http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html

[24] S. Song, S. P. Lichtenberg, and J. Xiao, "SUN RGB-D: A RGB-D scene understanding benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 567–576.

[25] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, "Segmentation and recognition using structure from motion point clouds," in *Proc. Eur. Conf. Comput. Vis.* Marseille, France: Springer, 2008, pp. 44–57.

[26] 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.

[27] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 2881–2890.

[28] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," 2018, *arXiv:1802.02611*. [Online]. Available: https://arxiv.org/abs/1802.02611

[29] I. J. Goodfellow, M. Mirza, D. Xiao, A. Courville, and Y. Bengio, "An empirical investigation of catastrophic forgetting in gradient-based neural networks," 2013, *arXiv:1312.6211*. [Online]. Available: https://arxiv.org/abs/1312.6211

[30] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning transferable features with deep adaptation networks," 2015, *arXiv:1502.02791*. [Online]. Available: https://arxiv.org/abs/1502.02791

[31] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 12, pp. 2935–2947, Dec. 2018.

[32] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3320–3328.

[33] J. Kirkpatrick *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proc. Nat. Acad. Sci.*, vol. 114, no. 13, pp. 3521–3526, Mar. 2017.

[34] B. Singh and L. S. Davis, "An analysis of scale invariance in object detection—SNIP," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3578–3587.

[35] S. Song *et al.*, "An easy-to-hard learning strategy for within-image co-saliency detection," *Neurocomputing*, vol. 358, pp. 166–176, Sep. 2019.

[36] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnets: Hints for thin deep nets," 2014, *arXiv:1412.6550*. [Online]. Available: https://arxiv.org/abs/1412.6550

[37] W. Zhu, X. Xiang, T. D. Tran, G. D. Hager, and X. Xie, "Adversarial deep structured nets for mass segmentation from mammograms," in *Proc. IEEE 15th Int. Symp. Biomed. Imag. (ISBI)*, Apr. 2018, pp. 847–850.

[38] H. Jung, J. Ju, M. Jung, and J. Kim, "Less-forgetting learning in deep neural networks," 2016, *arXiv:1607.00122*. [Online]. Available: https://arxiv.org/abs/1607.00122

[39] Y. Li, N. Wang, J. Liu, and X. Hou, "Demystifying neural style transfer," 2017, *arXiv:1701.01036*. [Online]. Available: https://arxiv.org/abs/1701.01036

[40] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Mach. Learn.*, vol. 79, nos. 1–2, pp. 151–175, May 2010.

[41] H. Zhang *et al.*, "Context encoding for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 7151–7160.

[42] F. Yu, D. Wang, E. Shelhamer, and T. Darrell, "Deep layer aggregation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2403–2412.

[43] 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.

[44] O. Kupyn, V. Budzan, M. Mykhailych, D. Mishkin, and J. Matas, "Deblurgan: Blind motion deblurring using conditional adversarial networks," 2017, *arXiv:1711.07064*. [Online]. Available: https://arxiv.org/abs/1711.07064

[45] Q. Zhu, J. Mai, and L. Shao, "A fast single image haze removal algorithm using color attenuation prior," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 3522–3533, Nov. 2015.

[46] B. Cai, X. Xu, K. Jia, C. Qing, and D. Tao, "DehazeNet: An end-to-end system for single image haze removal," *IEEE Trans. Image Process.*, vol. 25, no. 11, pp. 5187–5198, Nov. 2016.

[47] H. Zhang and V. M. Patel, "Densely connected pyramid dehazing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 3194–3203.

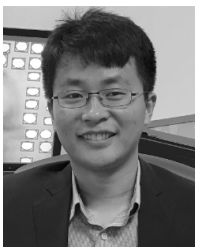
[48] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[49] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola, "Integrating structured biological data by kernel maximum mean discrepancy," *Bioinformatics*, vol. 22, no. 14, pp. e49–e57, Jul. 2006.

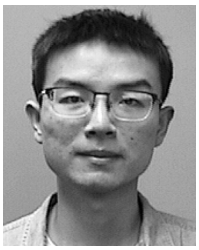
Dazhou Guo received the B.S degree in electronic engineering from the Dalian University of Technology, Dalian, China, in 2008, and the M.S. degree in information and informatics engineering from Tianjin University, Tianjin, China, in 2010. He is currently pursuing the Ph.D. degree in computer science with the University of South Carolina, USA. His research interests include computer vision, medical image processing, and machine learning.

Yanting Pei received the master's degree from the School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China, in 2015, where she is currently pursuing the Ph.D. degree. In 2018, she was a Visiting Ph.D. Student with the University of South Carolina, Columbia, SC, USA. Her current research interests include computer vision, machine learning, and deep learning.

Kang Zheng received the B.E. degree in electrical engineering from the Harbin Institute of Technology in 2012. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, University of South Carolina. His research interests include computer vision, image processing, and deep learning.



Hongkai Yu received the Ph.D. degree in computer science and engineering from the University of South Carolina, Columbia, SC, USA, in 2018. He then joined the Department of Computer Science, University of Texas-Rio Grande Valley, Edinburg, TX, USA, as an Assistant Professor. His research interests include computer vision, machine learning, deep learning, and intelligent transportation systems.



Yuhang Lu received the B.E. degree from the Chengdu University of Technology in 2013 and the M.E. degree from Wuhan University in 2015. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, University of South Carolina. His research interests include computer vision, machine learning, and image processing.

Song Wang (M'02–SM'13) received the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana–Champaign (UIUC), Champaign, IL, USA, in 2002. He was a Research Assistant with the Image Formation and Processing Group, Beckman Institute, UIUC, from 1998 to 2002. In 2002, he joined the Department of Computer Science and Engineering, University of South Carolina, Columbia, SC, USA, where he is currently a Professor. His current research interests include computer vision, image processing, and machine learning. He is currently a member of the IEEE Computer Society. He serves as the Publicity/Web Portal Chair for the Technical Committee of Pattern Analysis and Machine Intelligence of the IEEE Computer Society, an Associate Editor for the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, *Pattern Recognition Letters*, and *Electronics Letters*.