Deblur-YOLO: Real-Time Object Detection with Efficient Blind Motion Deblurring

Shen Zheng, Yuxiong Wu, Shiyu Jiang, Changjie Lu, Gaurav Gupta
College of Science and Technology
Wenzhou-Kean University, Wenzhou, China
{zhengsh,yuxiongwu,shiyujia,lucha,ggupta}@kean.edu

***Abstract*—Object detection has been a traditional yet open computer vision research field. In intensive studies, object detection models have achieved promising results regarding recognition accuracy and inference speed. However, previous state-of-the-art algorithms fail to operate at blurry images. In this work, we propose Deblur-YOLO, an efficient, YOLO-based and detection-driven approach robust to motion blur photographs. We introduce a generative adversarial network with a dilated feature pyramid generator, a pair of multi-scale discriminators with spectral normalization, and a detection discriminator. We design a new image quality metric called Smooth Peak Signal-to-Noise Ratio (SPSNR) for measuring the smoothness of the reconstructed image. Empirical studies on benchmark datasets demonstrate Deblur-YOLO's superiority. On COCO 2014, Set 5 and Set14, Deblur-YOLO achieves leading results for parameters, deblurring time, PSNR, SPSNR and SSIM. We also visually display the excellence of our deblurring performance to competing models.**

I. Introduction

Object detection had been a conventional yet challenging research field in computer vision. Object detection's purpose is to locate and recognize an instance of a specific class from digital photographs. It has been useful for various applications, including image classification, image captioning, pose estimation, face recognition, instance segmentation, and autonomous driving. [2]

Most traditional object detection algorithms leverage a pipeline of informative region selection, feature extraction, and object classification. Informative region selection uses a sliding window to go through the whole picture. Feature extraction methods like SIFT [3] and HOG [4] select representative features for the images. Meanwhile, classifiers like Support Vector Machine [5] and Random Forest [6] initiate a discriminative process that output the label. However, conventional methods have several significant drawbacks. First, it is inefficient to manually tune and move a sliding window through the entire image. Second, their limited model capacity cannot capture the high-level spatial features of the data.

Deep learning has become popular in computer vision tasks because of its excellent capability, accuracy, and flexibility. Various directions in object detection use deep learning. The pioneering researches exploit region proposals. R-CNN [7] leverages selective search algorithms to extract a fixed number of candidate regions called regional proposals. However, R-CNN is time-consuming and often generates suboptimal candidate regional proposals. Fast R-CNN [8] improves R-CNN by forwarding the whole image through a convolutional neural network (CNN). Then it crops and resizes the image features. Nevertheless, Fast R-CNN's computational efficiency degrades at finding appropriate regional proposals. Faster R-CNN [9] upgrades Fast R-CNN by constructing a separate region proposal network with CNNs. It also examines small regions instead of the complete image for object localization. Nonetheless, Faster R-CNN is still slow for real-time objection. Cascade R-CNN [10] addresses model degradation of Faster R-CNN with a growing IoU threshold at training and inference. It achieves this by proposing a multi-stage network that is sequentially more selective against negative predictions. If regional proposals algorithms could further optimize the model architecture, they could be more qualified for real-time detection tasks.

Single-stage methods aim toward accurate real-time object detection. Most one-stage approach performs detection as a classification or regression problem instead of using regional proposals. Yolo [11] is a leading research in this field. It uses a single CNN with bounding boxes and a class probability map to detect an object. The problem with Yolo is that it fails in detecting tiny objects. SSD [12] applies small convolutional filters to features maps with different scales and aspect ratios. The shortcoming of SSD is that it demands high-resolution layers for classifying small objects. However, those high-resolution layer contains multiple low-level features (e.g., background noise and edges) that downgrades the localization accuracy. RetinaNet [13] copes with that class imbalance problem. It introduces focal loss to mitigate the influence of easily classified background instances and to focus on vital positive examples. If single-stage methods could adopt multi-scale convolutional layers that efficiently address objects of different sizes, they could have achieved aligned accuracy with multi-stage approaches.

A significant limitation of the methods mentioned above is that they cannot deal with a degraded image. However, in real driving scenes, it is almost impossible to locate a clean traffic sign. Because of vehicle movement, camera shake, or suboptimal weather conditions, we often receive an image that is a blur. If object detection algorithms train with only ground-truth data, they fail to predict blurry photographs accurately.

To tackle that problem, we develop Deblur-Yolo, a novel object detection model that can address blurry image inputs. Unlike methods that separate image deblurring and object

(a) Clean Image (b) Blurred Image (c) Deblur-YOLO

Fig. 1: **Sample Detection Result**. Deblur-YOLO makes blur robust object detecton at a densely populated image from COCO 2014. Left: Yolov3 [1] at clean image. Middle: Yolov3 at Blurred Image. Right: Deblur-YOLO at Blurred Image

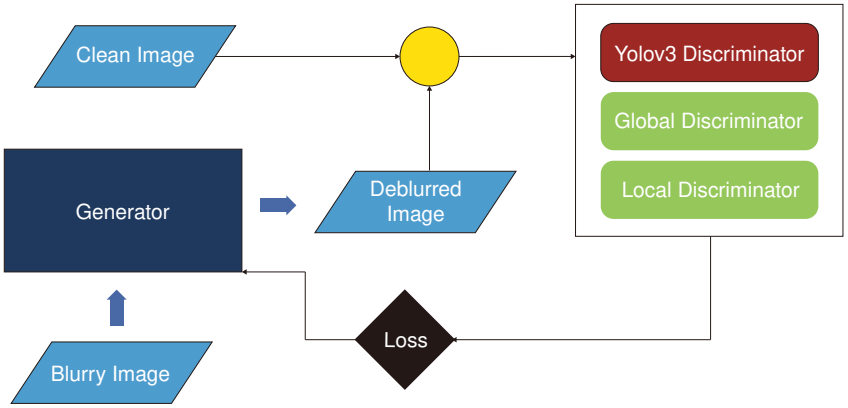

Fig. 2: **Model WorkFlow Design**. Deblur-YOLO is a Generative Adversarial Network (GAN) with one generator and a group of discriminators.

detection into two phrases, Deblur-Yolo is a one stage framework that performs deblurring and detection using a generative adversarial network [14]. First, we prepare pairs of clean and motion blur photos. Next, the generator convolve and upscale the blurry pictures. The discriminator receives the generated image and tries to distinguish it from the real images. During the min-max game, the two networks compete with each other. Finally, the generated sample should be similar to the ground truth data. The contributions of this paper summarize as below:

- We present an efficient object detection model that is robust to motion blur. To the best of our knowledge, we are the first to propose a one stage, detection-driven framework that integrates blind motion deblurring to real-time object detection.
- We introduce Dilated Feature Pyramid Network (DFPN), which utilizes dilated convolution [15] blocks to obtain a larger receptive field with less memory consumption.
- We design Smooth Peak Signal-to-Noise Ratio (SPSNR), which utilizes smooth l1 loss and effectively measures restored images' smoothness.
- We qualitatively and quantitatively demonstrate that our model achieves competitive performance against several state-of-the-art image deblurring models on COCO 2014, Set5, and Set14.

The rest of the paper is organized as following. Section II briefly review relevant literature. Section III introduces the proposed model. Section IV conducts the experiments and Section V derives the conclusion.

II. Related Works

A. Blind Motion Deblurring

Motion deblurring is an ill-posed task that can be solved either in a blind or non-blind way. Non-Blind deblurring methods presume an existing blur filter, whereas blind deblurring assumes no prior information about the blur. Traditional models usually target non-blind deblurring. Xu [17] proposes a sparse representation for natural language motion deblurring. Sun [18] proposes a patch prior to statistically synthetic structures. Deep Learning models, on the other hand, have made breakthroughs in blind image deblurring. Chakrabarti [19] forecasts the complex Fourier coefficients of a deconvolution kernel for motion deblurring, whereas Sun

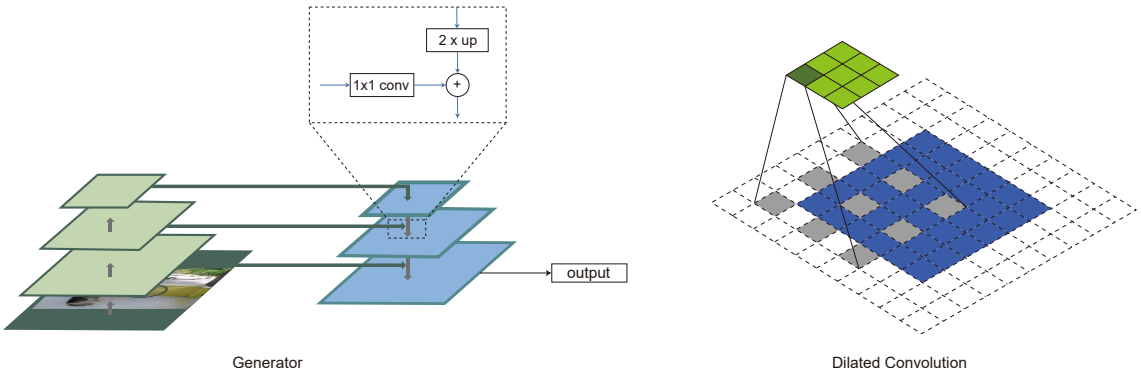

Fig. 3: **Generator Architecture**. Left: Generator building blocks with convolution and upsampling operations. Right: Dilated Convolution layers with stride 1, padding 2, dilation 2, kernel size 3 and filter number 128. We use blue, light green and dark green for dilated convolution blocks, vanilla convolution blocks and kernels, respectively. Each Convolution block consists of a convolution layer, a normalization layer and a ReLU [16] activation layer.

[20] utilizes CNN to approximate non-uniform motion blur kernel. Nah [21] proposes DeepDeblur, a kernel-free blind motion deblurring method that exploits multi-scale convolutional neural networks. Tao [22] introduces SRN-DeblurNet, which utilize a scaled recurrent neural network as a backbone to generate a more realistic deblurred image. DeblurGAN [23] and DeblurGANv2 [24] use generative adversarial network architecture with various backbones for better deblurring quality and efficiency. If the motion deblurring framework could consider the deblurring phrase's detection performance, they could have been more effective for detection at blurry scenes.

B. Generative Adversarial Network

Generative Adversarial Network (GAN) [14] defines a pair of competing players: The generator and the discriminator. The generator input with noise and try to generate an artificial sample to fool the discriminator, whereas the discriminator receives both the real and the generated samples and try to distinguish between them. The formulation of the min-max game is given as following.

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z))] \quad (1)$$

A particular type of GAN is conditional GAN [25]. Conditional GAN can generate samples according to specified labels and, therefore, become successful for image-to-image translation and image restoration tasks. Although GAN has achieved compelling performance in multifarious computer vision tasks, the vanilla GAN frameworks suffer from mode collapse and learning instability [26]. WGAN [26] replaces the JS divergence of GAN with Wasserstein loss and uses gradient clipping to enforce lip continuity. WGAN-GP [27] improves WGAN by replacing weight clipping with gradient regularization. LSGAN [28] leverages the least square loss function for the discriminator, thereby minimizing the Pearson divergence. Relativistic GAN [29] utilizes a relativistic discriminator that trains the generator to generate samples that is more realistic than stochastically sampled fake data. If GANs could adopt multiple generators or discriminators within one framework, they would have been able to balance between images of different scales.

C. Loss Function for Real-Time Object detection

Loss Function is essential for object detection because it helps detection models optimize their parameters effectively and efficiently. Single-stage methods are preferable for real-time object detection because of their fast inference. Yolo [11] uses a weighted average of localization loss, classification loss, and confidence loss. The classification loss reflects the class conditional probabilities for detected objects. The localization loss measures the mismatch for bounding box sizes and locations. And the confidence loss studies if an object is detected inside the box. Single Shot Detector (SSD) [12] leverages predictions from multiple end-layer feature maps to handle items of different sizes. It sets localization loss weight, ignoring negative matches and closing the positive prediction to the ground truth. RetinaNet [13] adopts Focal Loss to reduce the negative influences from well-classified samples (e.g., photograph background). Focal Loss is a modified cross-entropy loss that could concentrate model learning on hard negative samples. In this way, RetinaNet addressing the class balance during training. If detection loss could integrate with image quality measures, real-time object detection results would have been more similar to human perceptions.

III. Methodology

A. Model Workflow

We present our model workflow design in Fig.2. During training time, the generator receives images corrupted by motion blur and outputs the reconstructed photographs. The clean and deblurred images then go to the discriminators. The Yolo discriminator and the global discriminator use the complete image pairs, whereas the local discriminator uses randomly cropped ones. Three discriminators send the loss to the generator for it to optimize. We use pretrained Yolov3 as a discriminator, and we freeze its backpropagation. On the

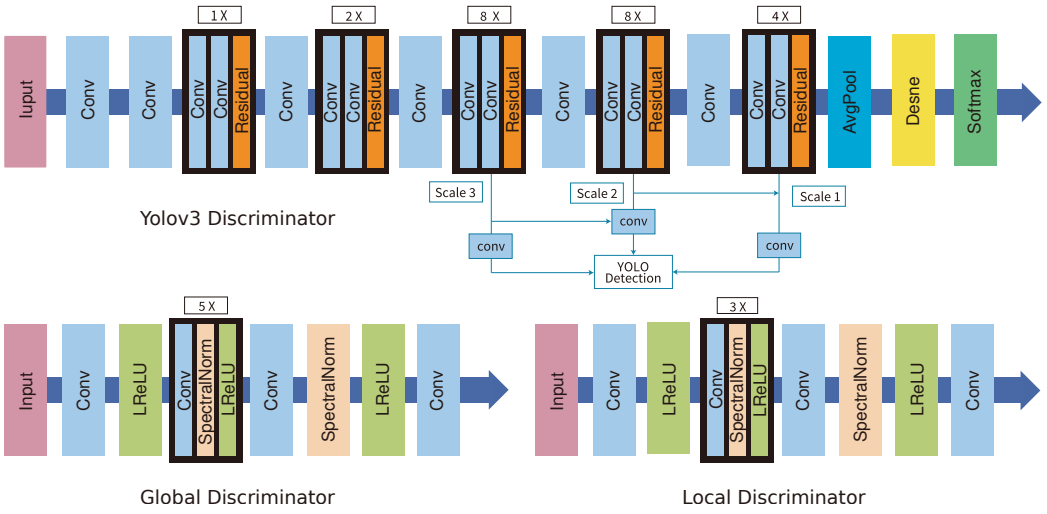

Fig. 4: **Discriminator Architecture**. Up: Detection Discriminator. Lower Left: Global-Scale Discriminator. Lower Right: Local-Scale Discriminator. For Yolov3 Discriminator, we use "Conv" for convolution blocks and "Residual" for residual blocks [30]. For Global and Local Discriminator, we use "Conv" for convolution layers and "LReLU" for Leaky ReLU [31] activation layers. Each convolution layers have stride 2, padding 2 and kernel size 4.

other hand, the global and the local discriminator return loss and update their parameters. During testing time, we close the global and local discriminator. Then we use the generator for deblurring and the Yolo discriminator for detection.

B. Model Architecture

Deblur-YOLO consists of a generator (Fig.3) and discriminators (Fig.4). For the generator, Deblur-YOLO utilizes a Dilated Feature Pyramid Network (DFPN), a feature pyramid network [32] with dilated convolution for the top-down pathway. The Feature Pyramid Network design allows capturing and deblurring small objects for better detection. The added dilated convolution layers enlarge the receptive field and reduce the memory consumption during the convolution process. Our generator consists of a bottom-up and top-down approach. The bottom-up pathway use pretrained MobileNetv2 [33] as backbone and has three convolution blocks with Batch Normalization [34]. Each convolution block has one lateral connection with 1×1 convolution. The illustration of the dilated convolution layers is in Fig.3. The top-down pathway receives the lateral inputs and combines it with $2 \times$ nearest neighbor upsample layers. The output is then sent to the dilated convolution blocks. We choose pretrained Yolov3 as the detection discriminator and design a global discriminator and a local discriminator. The local discriminator is similar to the global but has fewer convolution blocks. To stabilize GAN training, we adopts Spectral Normalization [35].

C. Loss Function

The loss function for our model consists of four parts: the detection loss from the YOLO discriminator, the adversarial loss from the global and the local discriminator, the content loss and the perceptual loss [36]. Content loss L_C is the l2 loss from the difference of the deblurred and the clean images, whereas perceptual loss L_P is the Euclidean loss on the third layer of VGG19 [37] feature maps. The adversarial loss L_A is from relativistic least square generative adversarial network [29] and the detection loss L_D is from Yolo [11]. The detailed loss function, including total loss L_G, discriminator adversarial loss L_{AD} and generator adversarial loss L_{AG} is given below.

$$\mathrm{L_G} = \lambda_1 * \mathrm{L_C} + \lambda_2 * \mathrm{L_P} + \lambda_3 * \mathrm{L_{A_G}} + \lambda_4 * \mathrm{L_D} \tag{2}$$

$$\begin{aligned}\mathrm{L_{A_G}} = \mathbb{E}_{z\sim p_z(z)}[(G(z) - \mathbb{E}_{x\sim p_{\text{data}}(x)}G(x) - 1)^2] \\ +\mathbb{E}_{x\sim p_{\text{data}}(x)}[(G(x) - \mathbb{E}_{z\sim p_z(z)}G(z) + 1)^2]\end{aligned} \tag{3}$$

$$\begin{aligned}\mathrm{L_{A_D}} = \mathbb{E}_{x\sim p_{\text{data}}(x)}[(D(x) - \mathbb{E}_{z\sim p_z(z)}D(G(z)) - 1)^2] \\ +\mathbb{E}_{z\sim p_z(z)}[(D(G(z)) - \mathbb{E}_{x\sim p_{\text{data}}(x)}D(x) + 1)^2]\end{aligned} \tag{4}$$

Inspired by [21, 23, 24, 38], we set λ_1 to 0.5, λ_2 to 0.006, λ_3 to 0.01 and λ_4 to 0.1.

IV. Experiments

A. Experimental Setup

In this section, we evaluate our model for motion deblurring and object detection against state-of-the-art models.

TABLE I: Comparison of mAP Score at COCO 2014

	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Clean Image	58.5	73.7	46.9	44.4	40.6	42.3	75.2	58.5	73.0	39.8	52.3	41.4	73.2	77.5	61.7	69.1	42.4	59.9	52.1	75.4	69.9
Blur Image	29.7	43.3	16.9	15.3	14.8	10.4	51.8	34.5	40.4	13.5	12.6	26.9	31.7	30.9	28.2	42.8	19.0	23.7	33.7	57.6	45.4
DeepDeblur	51.7	64.9	36.9	35.2	35.3	32.2	73.1	53.7	70.3	33.9	40.8	40.1	59.6	68.1	51.9	65.3	35.2	46.8	48.5	72.8	68.4
DynamicDeblur	**56.0**	**70.6**	**43.2**	**41.4**	**41.4**	**36.8**	**75.3**	**57.1**	**72.6**	**36.6**	**45.8**	40.0	**68.2**	**72.7**	**56.3**	**67.0**	**39.8**	**58.4**	49.9	**75.3**	**71.7**
SRN	52.3	70.1	38.2	35.8	35.8	31.8	71.9	53.4	69.2	33.1	39.4	39.8	63.4	66.6	53.5	64.1	35.1	51.7	48.0	**75.3**	69.0
DeblurGANv2(I-R)	42.0	55.0	28.6	26.6	30.2	24.9	61.4	44.9	53.5	27.5	35.4	32.4	47.4	53.7	39.6	51.8	24.8	41.2	39.2	65.2	55.9
DeblurGANv2(M)	40.8	52.2	27.4	25.0	28.9	24.3	61.0	44.3	53.7	25.9	31.7	30.5	45.2	49.4	39.2	50.8	25.0	38.6	40.6	66.0	56.8
Deblur-Yolo	47.5	55.5	33.8	30.0	37.7	29.7	67.7	51.1	62.6	31.2	39.5	**41.2**	51.4	54.7	44.9	56.1	33.6	53.9	**50.2**	72.8	52.2

(a) Clean Image (b) Blurred Image (c) DeepDeblur (d) SRN Deblur

(e) DynamicDeblur (f) DeblurGANv2(I-R) (g) DeblurGANv2(M) (h) Deblur-YOLO

Fig. 5: **Deblurring Result Comparison at Baby Picture from Set 5**

TABLE II: Deblurring Performance Comparison at COCO 2014

	Time	Params	PSNR	SPSNR	SSIM
Blur Image	None	None	21.02	116.51	0.701
DeepDeblur	1.5495	47.4	24.86	105.08	0.823
DynamicDeblur	1.5247	47.8	**27.19**	113.20	**0.873**
SRN	0.3790	86.9	24.61	99.92	0.815
DeblurGANv2(I-R)	0.1589	233.0	20.29	108.45	0.687
DeblurGANv2(M)	**0.0769**	**12.8**	20.34	124.94	0.687
Deblur-Yolo	0.0772	12.9	23.94	**131.39**	0.817

For model training, we use a benchmark dataset called COCO 2014 [39]. We apply the Pytorch [40] framework with 4 Nvidia GTX 1080 Ti GPU for training and 1 for testing. We train our Deblur-Yolo model from scratch for 50 epochs, using Adam [41] optimizer with its default setting. We warm up the learning rate for the first 5 epochs to mitigate the possibly poor initialization at the loss surface. We then use the learning rate at 5e-5 for 15 epochs, followed by another 30 epochs with a linear decay to 1e-7. Besides, We choose batch size as 16 and clip the gradient value within -1.99 to 1.99. It takes about 4 days for our model to converge.

B. Data Preparation

The Microsoft COCO 2014 [39] dataset has 80k+ training images and 40k+ validation images with bounding boxes and labels. We pad COCO images, which have a resolution less than 256, so that both our global and local discriminator shall function. After that, we apply motion blur to all COCO images to generate a blurred dataset.

C. Compared Models

We compare our models against a list of state-of-the-arts. Since traditional methods are too slow at inference for real-time detection tasks, we only compare with deep learning approaches. The competing models include DeepDeblur [21], SRN [22], DynamicDeblur [42], DeblurGANv2-InceptionResNet (DeblurGANv2(I-R)) and DeblurGANv2-MobileNet (DeblurGANv2(M)) [24]. To ensure fairness in comparison and approximate real-world scenarios, we calculate the inference time as the average processing time per image on a single Nvidia GTX 1080 Ti GPU.

(a) Clean Image (b) Blurred Image (c) DeepDeblur (d) SRN Deblur

(e) DynamicDeblur (f) DeblurGANv2(I-R) (g) DeblurGANv2(M) (h) Deblur-YOLO

Fig. 6: **Deblurring Result Comparison at Lenna Picture from Set14**

(a) clean Image (b) Blurred Image (c) DeepDeblur

(d) SRN Deblur (e) DynamicDeblur (f) DeblurGANv2(I-R)

(g) DeblurGANv2(M) (h) Deblur-YOLO

Fig. 7: **Deblurring and Detection Result Comparison at COCO 2014**

TABLE III: Deblurring Performance Comparison at Set 5 & Set 14

		Blur Image	DeepDeblur	DynamicDeblur	SRN	DeblurGANv2(I-R)	DeblurGANv2(M)	Deblur-Yolo
Set 5	PSNR	24.20	28.36	29.10	28.07	26.64	27.06	**29.39**
	SPSNR	113.79	104.80	113.99	98.77	103.74	122.66	**128.40**
	SSIM	0.66	0.81	0.85	0.80	0.74	0.77	**0.88**
Set 14	PSNR	23.12	26.65	27.35	25.90	25.95	25.03	**27.85**
	SPSNR	119.26	111.00	115.09	111.58	116.26	**128.30**	121.70
	SSIM	0.55	0.69	0.73	0.67	0.68	0.65	**0.75**

D. Evaluation Metrics

We use several benchmark metrics for assessing model performances. To balance between image deblurring and object detection performance, we prepare two sets of metrics. The objection detection metric is the mean average precision (mAP) for different prediction classes. The deblurring metric includes peak signal-to-noise ratio (PSNR) and structure similarity (SSIM). We additionally introduce a novel image quality evaluation metrics called Smooth peak signal-to-noise ratio (SPSNR). SPSNR is calculated by replacing l2 loss in PSNR with Smooth L1 loss. It is more stable than the l1 approach and is more balanced at measuring image smoothness than the l2 method. A good value for PSNR, SPSNR and SSIM should be 25, 120 and 0.8, respectively.

E. Quantitative Evaluation

We conduct a quantitative comparison of our method with other state-of-the-arts. Table I shows mAP scores with different deblurring models using Yolov3 as the detection backbone. Table II shows deblurring time in seconds, PSNR, SPSNR, and SSIM on COCO dataset and Table III Show the deblurring performance in terms of PSNR, SPSNR, and SSIM on Set5 and Set14 dataset. For all tables, the **bold** and blue points indicate the best and second best model performance, respectively. In the table, Params refers to the deblurring model parameters in MB, whereas Time refers to the deblurring time for a single image in seconds.

In Table I, comparing real-time deblurring models such as DeblurGANv2(I-R) and DeblurGANv2(M) with Deblur-YOLO, the accuracy of average mAP score increased 5.5% to 6.7%. Also, the mAP score is best for table and sofa; and the second-best score for boat, sheep, and train. Other methods such as DeepDeblur, DynamicDeblur and SRN have high mAP score nevertheless low SPSNR than the Deblur-YOLO (see Tablev II)

Table II shows that Deblur-YOLO surpasses all other models for SPSNR and competing with DeepDeblur and SRN for PSNR. Also, Deblur-YOLO outperforms SRN and is close to DeepDeblur for SSIM. The Deblur-YOLO is 2-20 times faster than most competing models. DeblurGANv2(M) has same speed as Deblur-YOLO however it doesn't perform good on other evaluation metrics.

Table III shows that Deblur-YOLO outperforms all other models for SPSNR and competing with DeepDeblur and SRN for PSNR. Also, Deblur-YOLO outperforms SRN and is close to DeepDeblur for SSIM. The Deblur-YOLO is 2-20 times faster than most competing models. DeblurGANv2(M) has same speed as Deblur-YOLO however it doesn't perform good on other evaluation metrics.

F. Qualitative Evaluation

We also qualitatively compare our model's performance against competing ones in this subsection. We first display the deblurring of different models. The baby image (Fig.5) from Set 5 and the Lenna image (Fig.6) from Set 14 is selected for comparison. In Fig.5, Deblur-YOLO has the best visual quality with the least artifacts among all competing models. In Fig.6, Deblur-YOLO is more realistic and smooth than all models except DeblurGAN(M). However, DeblurGAN(M) creates a distorted human face with unpleasant artifacts.

The deblurring and detection performance on the COCO dataset is in Fig.7. In this group of the image, we use Yolov3 as detection backbone. Each resulting image has bounding boxes of different colors denoted with captions. The Yolov3 discriminator has excellent detection performance at the clean image but significantly degrades at the blurred photograph. All models fail to recognize the closing truck and car in the lower right corner. DynamicDeblur is the only model that can realize the pedestrian in the distance on the left side. In comparison, Deblur-YOLO successfully detects the truck and highly overlapped persons in the middle of the photo. It also locates small objects like backpacks and bicycles.

V. Conclusion

This paper has introduced Deblur-YOLO, an efficient one-stage framework that integrates motion deblurring and object detection. We developed a generative adversarial network (GAN) with a dilated feature pyramid generator and a group of detection-driven, multi-scale discriminators. We also designed a more balanced image quality metric called Smooth Peak Signal-to-Noise Ratio (SPSNR). Our model, Deblur-YOLO, achieved leading scores and best visual quality at Set 5 and Set 14. On COCO 2014 , Deblur-YOLO attains real-time speed with competitive deblurring performance and less parameters to non-real-time methods. In the future, we hope to incorporate model compression techniques and other detection discriminators (e.g. SSD [12]). We also plan to explore task-driven deblurring with instance segmentation and semantic segmentation.

Acknowledgement

This work is supported by Wenzhou-Kean University with project number SpF2021011. We are grateful for Dr.Kennedy

E. Ehimwenma for his helpful comments on model annotation and result interpretation. We also thank Yayun Chen for her advice on model design.

REFERENCES

[1] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[2] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11):3212–3232, 2019.

[3] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[4] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005.

[5] Johan AK Suykens and Joos Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999.

[6] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[7] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[8] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[9] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016.

[10] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6154–6162, 2018.

[11] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[12] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[13] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27:2672–2680, 2014.

[15] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[16] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

[17] Li Xu, Shicheng Zheng, and Jiaya Jia. Unnatural l0 sparse representation for natural image deblurring. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1107–1114, 2013.

[18] Libin Sun, Sunghyun Cho, Jue Wang, and James Hays. Edge-based blur kernel estimation using patch priors. In *IEEE International Conference on Computational Photography (ICCP)*, pages 1–8. IEEE, 2013.

[19] Ayan Chakrabarti. A neural approach to blind motion deblurring. In *European conference on computer vision*, pages 221–235. Springer, 2016.

[20] Jian Sun, Wenfei Cao, Zongben Xu, and Jean Ponce. Learning a convolutional neural network for non-uniform motion blur removal. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 769–777, 2015.

[21] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3883–3891, 2017.

[22] Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiaya Jia. Scale-recurrent network for deep image deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8174–8182, 2018.

[23] Orest Kupyn, Volodymyr Budzan, Mykola Mykhailych, Dmytro Mishkin, and Jiří Matas. Deblurgan: Blind motion deblurring using conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8183–8192, 2018.

[24] Orest Kupyn, Tetiana Martyniuk, Junru Wu, and Zhangyang Wang. Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8878–8887, 2019.

[25] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[26] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[27] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777, 2017.

[28] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2794–2802, 2017.

[29] Alexia Jolicoeur-Martineau. The relativistic discriminator: a key element missing from standard gan. *arXiv preprint arXiv:1807.00734*, 2018.

[30] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[31] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3. Citeseer, 2013.

[32] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[33] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[34] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[35] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

[36] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.

[37] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[38] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[39] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[40] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019.

[41] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[42] Hongyun Gao, Xin Tao, Xiaoyong Shen, and Jiaya Jia. Dynamic scene deblurring with parameter selective sharing and nested skip connections. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3848–3856, 2019.