DeblurGAN: Blind Motion Deblurring Using Conditional Adversarial Networks

Orest Kupyn[1,3], Volodymyr Budzan[1,3], Mykola Mykhailych[1], Dmytro Mishkin[2], Jiři Matas[2]

[1] Ukrainian Catholic University, Lviv, Ukraine
{kupyn, budzan, mykhailych}@ucu.edu.ua

[2] Visual Recognition Group, Center for Machine Perception, FEE, CTU in Prague
{mishkdmy, matas}@cmp.felk.cvut.cz

[3] ELEKS Ltd.

Abstract

We present DeblurGAN, an end-to-end learned method for motion deblurring. The learning is based on a conditional GAN and the content loss . DeblurGAN achieves state-of-the art performance both in the structural similarity measure and visual appearance. The quality of the deblurring model is also evaluated in a novel way on a real-world problem – object detection on (de-)blurred images. The method is 5 times faster than the closest competitor – DeepDeblur [25]. We also introduce a novel method for generating synthetic motion blurred images from sharp ones, allowing realistic dataset augmentation.

The model, code and the dataset are available at https://github.com/KupynOrest/DeblurGAN

1. Introduction

This work is on blind motion deblurring of a single photograph. Significant progress has been recently achieved in related areas of image super-resolution [20] and inpainting [45] by applying generative adversarial networks (GANs) [10]. GANs are known for the ability to preserve texture details in images, create solutions that are close to the real image manifold and look perceptually convincing. Inspired by recent work on image super-resolution [20] and image-to-image translation by generative adversarial networks [16], we treat deblurring as a special case of such image-to-image translation. We present DeblurGAN – an approach based on conditional generative adversarial networks [24] and a multi-component loss function. Unlike previous work we use Wasserstein GAN [2] with the gradient penalty [11] and perceptual loss [17]. This encourages solutions which are perceptually hard to distinguish from real sharp images and allows to restore finer texture details than if using traditional MSE or MAE as an optimization target.

Figure 1: DeblurGAN helps object detection. YOLO [30] detections on the blurred image (top), the DeblurGAN restored (middle) and the sharp ground truth image from the GoPro [25] dataset.

Figure 2: GoPro images [25] processed by DeblurGAN. Blurred – left, DeblurGAN – center, ground truth sharp – right.

We make three contributions. First, we propose a loss and architecture which obtain state-of-the art results in motion deblurring, while being 5x faster than the fastest competitor. Second, we present a method based on random trajectories for generating a dataset for motion deblurring training in an automated fashion from the set of sharp image. We show that combining it with an existing dataset for motion deblurring learning improves results compared to training on real-world images only. Finally, we present a novel dataset and method for evaluation of deblurring algorithms based on how they improve object detection results.

2. Related work

2.1. Image Deblurring

The common formulation of non-uniform blur model is the following:

$$I_B = k(M) * I_S + N, \tag{1}$$

where I_B is a blurred image, $k(M)$ are unknown blur kernels determined by motion field M. I_S is the sharp latent image, $*$ denotes the convolution, N is an additive noise. The family of deblurring problems is divided into two types: blind and non-blind deblurring. Early work [37] mostly focused on non-blind deblurring, making an assumption that the blur kernels $k(M)$ are known. Most rely on the classical Lucy-Richardson algorithm, Wiener or Tikhonov filter to perform the deconvolution operation and obtain I_S estimate. Commonly the blur function is unknown, and blind deblurring algorithms estimate both latent sharp image I_S and blur kernels $k(M)$. Finding a blur function for each pixel is an ill-posed problem, and most of the existing algorithms rely on heuristics, image statistics and assumptions on the sources of the blur. Those family of methods addresses the blur caused by camera shake by considering blur to be uniform across the image. Firstly, the camera motion is estimated in terms of the induced blur kernel, and then the effect is reversed by performing a deconvolution operation. Starting with the success of Fergus *et al*. [8], many methods [44][42][28][3] has been developed over the last ten years. Some of the methods are based on an iterative approach [8] [44], which improve the estimate of the motion kernel and sharp image on each iteration by using parametric prior models. However, the running time, as well as the stopping criterion, is a significant problem for those kinds of algorithms. Others use assumptions of a local linearity of a blur function and simple heuristics to quickly estimate the unknown kernel. These methods are fast but work well on a small subset of images.

Recently, Whyte *et al*. [40] developed a novel algorithm

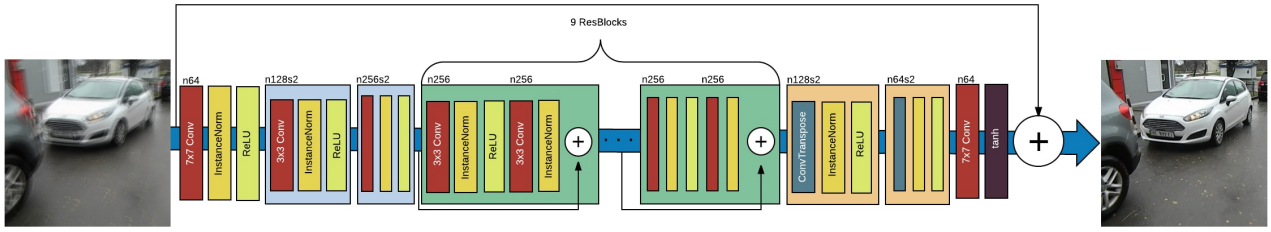


Figure 3: DeblurGAN generator architecture. DeblurGAN contains two strided convolution blocks with stride $\frac{1}{2}$, nine residual blocks [13] and two transposed convolution blocks. Each ResBlock consists of a convolution layer, instance normalization layer, and ReLU activation.

for non-uniform blind deblurring based on a parametrized geometric model of the blurring process in terms of the rotational velocity of the camera during exposure. Similarly Gupta *et al*. [12] made an assumption that the blur is caused only by 3D camera movement. With the success of deep learning, over the last few years, there appeared some approaches based on convolutional neural networks (CNNs). Sun *et al*. [36] use CNN to estimate blur kernel, Chakrabarti [6] predicts complex Fourier coefficients of motion kernel to perform non-blind deblurring in Fourier space whereas Gong [9] use fully convolutional network to move for motion flow estimation. All of these approaches use CNN to estimate the unknown blur function. Recently, a kernel-free end-to-end approaches by Noorozi [27] and Nah [25] that uses multi-scale CNN to directly deblur the image. Ramakrishnan *et al*. [29] use the combination of pix2pix framework [16] and densely connected convolutional networks [15] to perform blind kernel-free image deblurring. Such methods are able to deal with different sources of the blur.

2.2. Generative adversarial networks

The idea of generative adversarial networks, introduced by Goodfellow *et al*. [10], is to define a game between two competing networks: the discriminator and the generator. The generator receives noise as an input and generates a sample. A discriminator receives a real and generated sample and is trying to distinguish between them. The goal of the generator is to fool the discriminator by generating perceptually convincing samples that can not be distinguished from the real one. The game between the generator G and discriminator D is the minimax objective:

$$\min_G \max_D \mathop{\mathbb{E}}_{x \backsim \mathbb{P}_r} [\log(D(x))] + \mathop{\mathbb{E}}_{\tilde{x} \backsim \mathbb{P}_g} [\log(1 - D(\tilde{x}))] \quad (2)$$

where $\mathbb{P}_r$ is the data distribution and $\mathbb{P}_g$ is the model distribution, defined by $\tilde{x} = G(z), z \backsim P(z)$, the input z is a sample from a simple noise distribution. GANs are known for its ability to generate samples of good perceptual quality, however, training of vanilla version suffer from many problems such as mode collapse, vanishing gradients etc, as described in [33]. Minimizing the value function in GAN is equal to minimizing the Jensen-Shannon divergence between the data and model distributions on x. Arjovsky *et al*. [2] discuss the difficulties in GAN training caused by JS divergence approximation and propose to use the Earth-Mover (also called Wasserstein-1) distance $W(q, p)$. The value function for WGAN is constructed using Kantorovich-Rubinstein duality [39]:

$$\min_G \max_{D \in \mathcal{D}} \mathop{\mathbb{E}}_{x \backsim \mathbb{P}_r} [D(x)] - \mathop{\mathbb{E}}_{\tilde{x} \backsim \mathbb{P}_g} [D(\tilde{x})] \quad (3)$$

where $\mathcal{D}$ is the set of $1-$Lipschitz functions and $\mathbb{P}_g$ is once again the model distribution The idea here is that critic value approximates $K \cdot W(P_r, P_\theta)$, where K is a Lipschitz constant and $W(P_r, P_\theta)$ is a Wasserstein distance. In this setting, a discriminator network is called critic and it approximates the distance between the samples. To enforce Lipschitz constraint in WGAN Arjovsky *et al*. add weight clipping to $[-c, c]$. Gulrajani *et al*. [11] propose to add a gradient penalty term instead:

$$\lambda \mathop{\mathbb{E}}_{\tilde{x} \backsim \mathbb{P}_{\tilde{x}}} [(\|\nabla_{\tilde{x}} D(\tilde{x})\|_2 - 1)^2] \quad (4)$$

to the value function as an alternative way to enforce the Lipschitz constraint. This approach is robust to the choice of generator architecture and requires almost no hyperparameter tuning. This is crucial for image deblurring as it allows to use novel lightweight neural network architectures in contrast to standard Deep ResNet architectures, previously used for image deblurring [25].

2.3. Conditional adversarial networks

Generative Adversarial Networks have been applied to different image-to-image translation problems, such as super resolution [20], style transfer [22], product photo generation [5] and others. Isola *et al*. [16] provides a detailed overview of those approaches and present conditional GAN architecture also known as *pix2pix*. Unlike vanilla GAN,

cGAN learns a mapping from observed image x and random noise vector z, to $y : G : x, z \rightarrow y$. Isola *et al.* also put a condition on the discriminator and use U-net architecture [31] for generator and Markovian discriminator which allows achieving perceptually superior results on many tasks, including synthesizing photos from label maps, reconstructing objects from edge maps, and colorizing images.

3. The proposed method

The goal is to recover sharp image I_S given only a blurred image I_B as an input, so no information about the blur kernel is provided. Debluring is done by the trained CNN G_{θ_G}, to which we refer as the Generator. For each I_B it estimates corresponding I_S image. In addition, during the training phase, we introduce critic the network D_{θ_D} and train both networks in an adversarial manner.

3.1. Loss function

We formulate the loss function as a combination of content and adversarial loss:

$$\mathcal{L} = \underbrace{\underbrace{\mathcal{L}_{GAN}}_{adv\ loss} + \underbrace{\lambda \cdot \mathcal{L}_X}_{content\ loss}}_{total\ loss} \tag{5}$$

where the λ equals to 100 in all experiments. Unlike Isola *et al.* [16] we do not condition the discriminator as we do not need to penalize mismatch between the input and output. **Adversarial loss** Most of the papers related to conditional GANs, use vanilla GAN objective as the loss [20][25] function. Recently [47] provides an alternative way of using least aquare GAN [23] which is more stable and generates higher quality results. We use WGAN-GP [11] as the critic function, which is shown to be robust to the choice of generator architecture [2]. Our premilinary experiments with different architectures confirmed that findings and we are able to use architecture much lighter than ResNet152 [25], see next subsection. The loss is calculated as the following:

$$\mathcal{L}_{GAN} = \sum_{n=1}^{N} -D_{\theta_D}(G_{\theta_G}(I^B)) \tag{6}$$

DeblurGAN trained without GAN component converges, but produces smooth and blurry images.

Content loss. Two classical choices for "content" loss function are **L1** or **MAE** loss, **L2** or **MSE** loss on raw pixels. Using those functions as sole optimization target leads to the blurry artifacts on generated images due to the pixel-wise average of possible solutions in the pixel space [20]. Instead, we adopted recently proposed Perceptual loss [17]. Perceptual loss is a simple L2-loss, but based on the difference of the generated and target image CNN feature maps. It is defined as following:

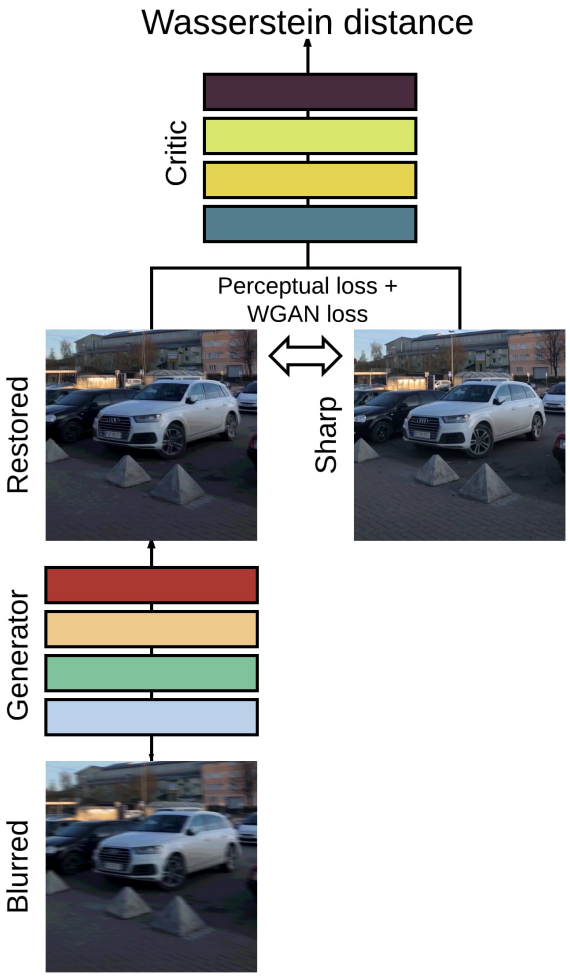


Figure 4: DeblurGAN training. The generator network takes the blurred image as input and produces the estimate of the sharp image. The critic network takes the restored and sharp images and outputs a distance between them. The total loss consists of the WGAN loss from critic and the perceptual loss [17]. The perceptual loss is the difference between the VGG-19 [34] *conv3.3* feature maps of the sharp and restored images. At test time, only the generator is kept.

$$\mathcal{L}_X = \frac{1}{W_{i,j} H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\phi_{i,j}(I^S)_{x,y} - \phi_{i,j}(G_{\theta_G}(I^B))_{x,y})^2$$

where $\phi_{i,j}$ is the feature map obtained by the j-th convolution (after activation) before the i-th maxpooling layer within the VGG19 network, pretrained on ImageNet [7], $W_{i,j}$ and $H_{i,j}$ are the dimensions of the feature maps. In our work we use activations from $VGG_{3,3}$ convolutional layer. The activations of the deeper layers represents the features of a higher abstraction [46][20]. The perceptual loss focuses on restoring general content [16] [20] while ad-

Figure 5: Examples of generated camera motion trajectory and the blur kernel and the corresponding blurred images.

versarial loss focuses on restoring texture details. DeblurGAN trained without Perceptual loss or with simple MSE on pixels instead doesn't converge to meaningful state.

Additional regularization. We have also tried to add TV regularization and model trained with it yields worse performance – 27.9 vs. 28.7 w/o PSNR on GoPro dataset.

3.2. Network architecture

Generator CNN architecture is shown in Figure 3. It is similar to one proposed by Johnson *et al*. [17] for style transfer task. It contains two strided convolution blocks with stride $\frac{1}{2}$, nine residual blocks [13] (ResBlocks) and two transposed convolution blocks. Each ResBlock consists of a convolution layer, instance normalization layer [38], and ReLU [26] activation. Dropout [35] regularization with a probability of 0.5 is added after the first convolution layer in each ResBlock. In addition, we introduce the global skip connection which we refer to as ResOut. CNN learns a residual correction I_R to the blurred image I_B, so $I_S = I_B + I_R$. We find that such formulation makes training faster and resulting model generalizes better. During the training phase, we define a critic network D_{θ_D}, which is Wasserstein GAN [2] with gradient penalty [11], to which we refer as WGAN-GP. The architecture of critic network is identical to PatchGAN [16, 22]. All the convolutional layers except the last are followed by InstanceNorm layer and LeakyReLU [41] with $\alpha = 0.2$.

4. Motion blur generation

There is no easy method to obtain image pairs of corresponding sharp and blurred images for training.A typical approach to obtain image pairs for training is to use a high frame-rate camera to simulate blur using average of sharp frames from video [27, 25]. It allows to create realistic blurred images but limits the image space only to scenes present in taken videos and makes it complicated to scale the dataset. Sun *et al*. [36] creates synthetically blurred images by convolving clean natural images with one out of 73 possible linear motion kernels, Xu *et al*. [43] also use linear motion kernels to create synthetically blurred images.

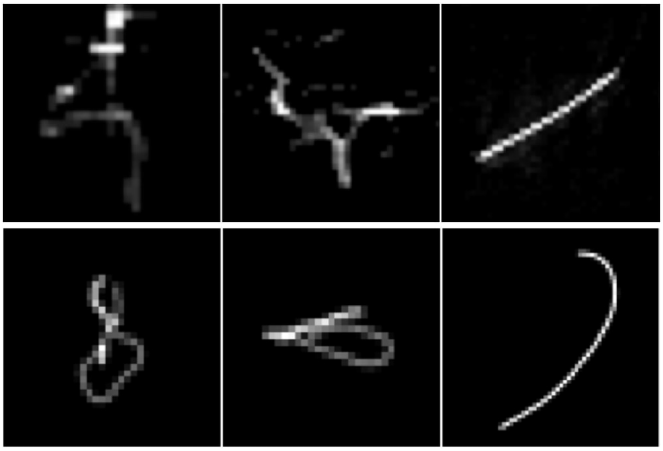


Figure 6: Top row: Blur kernels from real-world images estimated by Fergus *et al*. [8]. Bottom row: Synthetically generated kernels by our method. Our randomized method can simulate wide variety of realistic blur kernels with different level of non-linearity.

Chakrabarti [6] creates blur kernel by sampling 6 random points and fitting a spline to them. We take a step further and propose a method, which simulates more realistic and complex blur kernels. We follow the idea described by Boracchi and Foi [4] of random trajectories generation. Then the kernels are generated by applying sub-pixel interpolation to the trajectory vector. Each trajectory vector is a complex valued vector, which corresponds to the discrete positions of an object following 2D random motion in a continuous domain. Trajectory generation is done by Markov process, summarized in Algorithm 1. Position of the next point of the trajectory is randomly generated based on the previous point velocity and position, gaussian perturbation, impulse perturbation and deterministic inertial component.

5. Training Details

We implemented all of our models using PyTorch[1] deep learning framework. The training was performed on a single Maxwell GTX Titan-X GPU using three

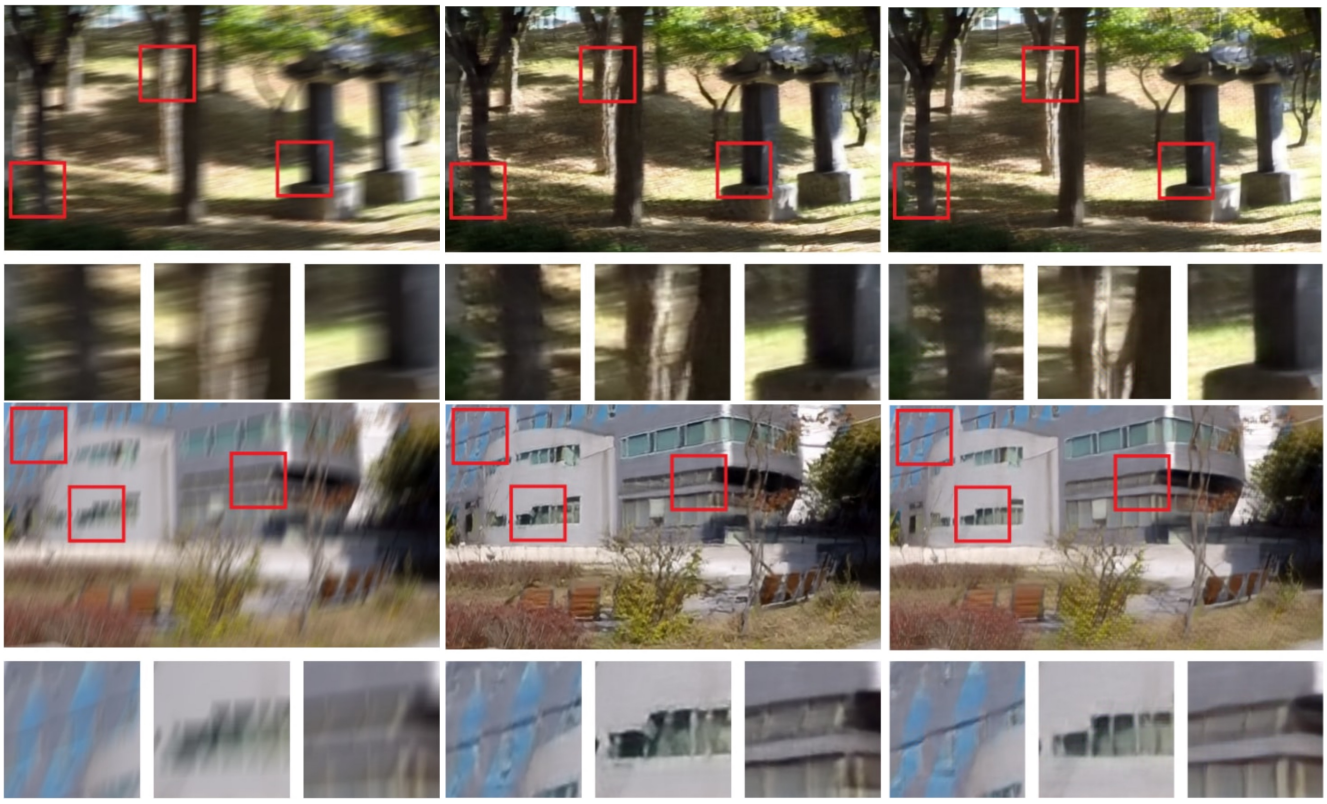


Figure 7: Results on the GoPro test dataset. From left to right: blurred photo, Nah *et al*. [25], DeblurGAN.

Figure 8: Results on the Kohler dataset. From left to right: blurred photo, Nah *et al*. [25], DeblurGAN.

different datasets. The first model to which we refer as $DeblurGAN_{WILD}$ was trained on a random crops of size 256x256 from 1000 GoPro training dataset images [25] downscaled by a factor of two. The second one $DeblurGAN_{Synth}$ was trained on 256x256 patches from MS COCO dataset blurred by method, presented in previous

Algorithm 1 Motion blur kernel generation.

Parameters:
M = 2000 – number of iterations,
L_{max} = 60 – max length of the movement,
p_s = 0.001 – probability of impulsive shake,
I – inertia term, uniform from (0,0.7),
p_b – probability of big shake, uniform from (0,0.2),
p_g – probability of gaussian shake, uniform from (0,0.7),
ϕ – initial angle, uniform from (0,2π),
x – trajectory vector.

procedure BLUR(Img, M, L_{max}, p_s)
  $v_0 \leftarrow cos(\phi) + sin(\phi) * i$
  $v \leftarrow v_o * L_{max}/(M-1)$
  $x = \text{zeros}(M, 1)$
  for $t = 1$ to $M - 1$ **do**
    if randn $< p_b * p_s$ **then**
      nextDir $\leftarrow 2 \cdot v \cdot e^{i*(\pi+(\text{randn}-0.5))})$
    else:
      nextDir $\leftarrow 0$
    $dv \leftarrow \text{nextDir} + p_s * (p_g * (\text{randn} + i * \text{randn}) * I * x[t] * (L_{max}/(M-1))$
    $v \leftarrow v + dv$
    $v \leftarrow (v/abs(v)) * L_{max}/(M-1)$
    $x[t+1] \leftarrow x[t] + v$
  Kernel $\leftarrow$ sub pixel interpolation(x)
  Blurred image $\leftarrow conv(\text{Kernel}, \text{Img})$
  return Blurred image

Section. We also trained *DeblurGAN*$_{Comb}$ on a combination of synthetically blurred images and images taken in the wild, where the ratio of synthetically generated images to the images taken by a high frame-rate camera is 2:1. As the models are fully convolutional and are trained on image patches they can be applied to images of arbitrary size. For optimization we follow the approach of [2] and perform 5 gradient descent steps on D_{θ_D}, then one step on G_{θ_G}, using Adam [18] as a solver. The learning rate is set initially to 10^{-4} for both generator and critic. After the first 150 epochs we linearly decay the rate to zero over the next 150 epochs. At inference time we follow the idea of [16] and apply both dropout and instance normalization. All the models were trained with a batch size = 1, which showed empirically better results on validation. The training phase took 6 days for training one *DeblurGAN* network.

6. Experimental evaluation

6.1. GoPro Dataset

GoPro dataset[25] consists of 2103 pairs of blurred and sharp images in 720p quality, taken from various scenes. We compare the results of our models with state of the art models [36], [25] on standard metrics and also show the running time of each algorithm on a single **GPU**. Results are in Table1. DeblurGAN shows superior results in terms of structured self-similarity, is close to state-of-the-art in peak signal-to-noise-ratio and provides better looking results by visual inspection. In contrast to other neural models, our network does not use L2 distance in pixel space so it is not directly optimized for PSNR metric. It can handle blur caused by camera shake and object movement, does not suffer from usual artifacts in kernel estimation methods and at the same time has more than 6x fewer parameters comparing to Multi-scale CNN , which heavily speeds up the inference. Deblured images from test on GoPro dataset are shown in Figure 7.

Table 1: Peak signal-to-noise ratio and the structural similarity measure, mean over the GoPro test dataset of 1111 images. All models were tested on the *linear* image subset. State-of-art results $(*)$ by Nah *et al.* [25] obtained on the *gamma* subset.

	Sun *et al.*	Nah *et al.*	Xu *et al.*	DeblurGAN		
Metric	[36]	[25]	[44]	*WILD*	*Synth*	*Comb*
PSNR	24.6	28.3/**29.1***	25.1	27.2	23.6	**28.7**
SSIM	0.842	0.916	0.89	0.954	0.884	**0.958**
Time	20 min	4.33 s	13.41 s	**0.85 s**		

6.2. Kohler dataset

Kohler dataset [19] consists of 4 images blurred with 12 different kernels for each of them. This is a standard benchmark dataset for evaluation of blind deblurring algorithms. The dataset is generated by recording and analyzing real camera motion, which is played back on a robot platform such that a sequence of sharp images is recorded sampling the 6D camera motion trajectory. Results are in Table 2, similar to GoPro evaluation.

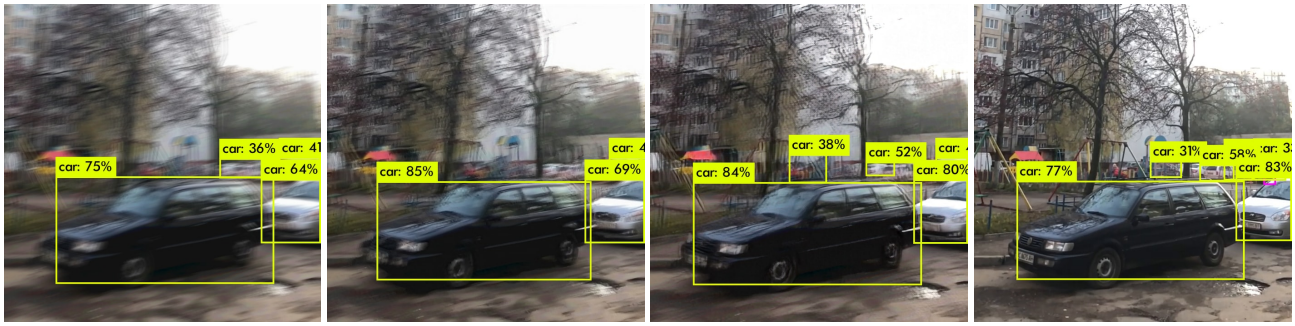
6.3. Object Detection benchmark on YOLO

Object Detection is one of the most well-studied problems in computer vision with applications in different domains from autonomous driving to security. During the last few years approaches based on Deep Convolutional Neural Networks showed state of the art performance comparing to traditional methods. However, those networks are trained on limited datasets and in real-world settings images are often degraded by different artifacts, including motion blur, Similar to [21] and [32] we studied the influence of motion blur on object detection and propose a new way to evaluate the quality of deblurring algorithm based on results of object detection on a pretrained YOLO [30] network.

For this, we constructed a dataset of sharp and blurred street views by simulating camera shake using a high frame-

Table 2: Peak signal-to-noise ratio and structural similarity measure, mean on the Kohler dataset. Xu *et al*. [44] and Whyte *et al*. [40] are non-CNN blind deblurring methods, whereas Sun *et al*. [36] and Nah *et al*. [25] use CNN.

Method	Sun *et al*.	Nah *et al*.	Xu *et al*.	Whyte *et al*.	DeblurGAN		
Metric	[36]	[25]	[44]	[40]	*WILD*	*Synth*	*Comb*
PSNR	25.22	26.48	**27.47**	27.03	26.10	25.67	25.86
SSIM	0.773	0.807	0.811	0.809	**0.816**	0.792	0.802



(a) Blurred photo (b) Nah *et al*. [25] (c) DeblurGAN (d) Sharp photo

Figure 9: YOLO object detection before and after deblurring

rate video camera. Following [14][25][27] we take a random between 5 and 25 frames taken by 240fps camera and compute the blurred version of a middle frame as an average of those frames. All the frames are gamma-corrected with $\gamma = 2.2$ and then the inverse function is taken to obtain the final blurred frame. Overall, the dataset consists of 410 pairs of blurred and sharp images, taken from the streets and parking places with different number and types of cars.

Blur source includes both camera shake and blur caused by car movement. The dataset and supplementary code are available online. Then sharp images are feed into the YOLO network and the result after visual verification is assigned as ground truth. Then YOLO is run on blurred and recovered versions of images and average recall and precision between obtained results and ground truth are calculated. This approach corresponds to the quality of deblurring models on real-life problems and correlates with the visual quality and sharpness of the generated images, in contrast to standard PSNR metric. The precision, in general, is higher on blurry images as there are no sharp object boundaries and smaller object are not detected as it shown in Figure 9.

Results are shown in Table 3. DeblurGAN significantly outperforms competitors in terms of recall and F1 score.

Table 3: Results of YOLO [30] object detection on blurred and restored photos using DeblurGAN and Nah *et al*. [25] algorithms. Results on corresponding sharp images are considered ground truth. DeblurGAN has higher recall and F1 score than its competitors.

Method	prec.	recall	F1 score
no deblur	0.821	0.437	0.570
Nah *et al*. [25]	**0.834**	0.552	0.665
DeblurGAN WILD	0.764	0.631	0.691
DeblurGAN synth	0.801	0.517	0.628
DeblurGAN comb	0.671	**0.742**	**0.704**

7. Conclusion

We described a kernel-free blind motion deblurring learning approach and introduced DeblurGAN which is a Conditional Adversarial Network that is optimized using a multi-component loss function. In addition to this, we implemented a new method for creating a realistic synthetic motion blur able to model different blur sources. We introduce a new benchmark and evaluation protocol based on results of object detection and show that DeblurGAN significantly helps detection on blurred images.

8. Acknowledgements

The authors were supported by the ELEKS Ltd., ARVI Lab, Czech Science Foundation Project GACR P103/12/G084, the Austrian Ministry for Transport, Innovation and Technology, the Federal Ministry of Science, Research and Economy, and the Province of Upper Austria in the frame of the COMET center, the CTU student grant SGS17/185/OHK3/3T/13. We thank Huaijin Chen for finding the bug in peak-signal-to-noise ratio evaluation.

References

[1] PyTorch. http://pytorch.org. 5

[2] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein GAN. *ArXiv e-prints*, Jan. 2017. 1, 3, 4, 5, 7

[3] S. D. Babacan, R. Molina, M. N. Do, and A. K. Katsaggelos. Bayesian blind deconvolution with general sparse image priors. In *European Conference on Computer Vision (ECCV)*, Firenze, Italy, October 2012. Springer. 2

[4] G. Boracchi and A. Foi. Modeling the performance of image restoration from motion blur. *Image Processing, IEEE Transactions on*, 21(8):3502 –3517, aug. 2012. 5

[5] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. Unsupervised Pixel-Level Domain Adaptation with Generative Adversarial Networks. *ArXiv e-prints*, Dec. 2016. 3

[6] A. Chakrabarti. A neural approach to blind motion deblurring. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016. 3, 5

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009. 4

[8] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman. Removing camera shake from a single photograph. *ACM Trans. Graph.*, 25(3):787–794, July 2006. 2, 5

[9] D. Gong, J. Yang, L. Liu, Y. Zhang, I. Reid, C. Shen, A. Van Den Hengel, and Q. Shi. From Motion Blur to Motion Flow: a Deep Learning Solution for Removing Heterogeneous Motion Blur. 2016. 3

[10] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Networks. June 2014. 1, 3

[11] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved Training of Wasserstein GANs. *ArXiv e-prints*, Mar. 2017. 1, 3, 4, 5

[12] A. Gupta, N. Joshi, C. L. Zitnick, M. Cohen, and B. Curless. Single image deblurring using motion density functions. In *Proceedings of the 11th European Conference on Computer Vision: Part I*, ECCV'10, pages 171–184, Berlin, Heidelberg, 2010. Springer-Verlag. 3

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015. 3, 5

[14] M. Hirsch, C. J. Schuler, S. Harmeling, and B. Scholkopf. Fast removal of non-uniform camera shake. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pages 463–470, Washington, DC, USA, 2011. IEEE Computer Society. 8

[15] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017. 3

[16] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *arxiv*, 2016. 1, 3, 4, 5, 7

[17] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, 2016. 1, 4, 5

[18] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 7

[19] R. Köhler, M. Hirsch, B. Mohler, B. Schölkopf, and S. Harmeling. Recording and playback of camera shake: Benchmarking blind deconvolution with a real-world database. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part VII*, ECCV'12, pages 27–40, Berlin, Heidelberg, 2012. Springer-Verlag. 7

[20] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. *ArXiv e-prints*, Sept. 2016. 1, 3, 4

[21] B. Li, X. Peng, Z. Wang, J. Xu, and D. Feng. An All-in-One Network for Dehazing and Beyond. *ArXiv e-prints*, July 2017. 7

[22] C. Li and M. Wand. Precomputed Real-Time Texture Synthesis with Markovian Generative Adversarial Networks. *ArXiv e-prints*, Apr. 2016. 3, 5

[23] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, and Z. Wang. Least squares generative adversarial networks, 2016. cite arxiv:1611.04076. 4

[24] M. Mirza and S. Osindero. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014. 1

[25] S. Nah, T. Hyun, K. Kyoung, and M. Lee. Deep Multi-scale Convolutional Neural Network for Dynamic Scene Deblurring. 2016. 1, 2, 3, 4, 5, 6, 7, 8

[26] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning (ICML)*, pages 807–814, 2010. 5

[27] M. Noroozi, P. Chandramouli, and P. Favaro. Motion Deblurring in the Wild. 2017. 3, 5, 8

[28] D. Perrone and P. Favaro. Total variation blind deconvolution: The devil is in the details. In *EEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 2

[29] S. Ramakrishnan, S. Pachori, A. Gangopadhyay, and S. Raman. Deep generative filter for motion deblurring. *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 2993–3000, 2017. 3

[30] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You Only Look Once: Unified, Real-Time Object Detection. *ArXiv e-prints*, June 2015. 1, 7, 8

[31] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. *ArXiv e-prints*, May 2015. 4

[32] M. S. M. Sajjadi, B. Schölkopf, and M. Hirsch. Enhancenet: Single image super-resolution through automated texture synthesis. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 4501–4510, 2017. 7

[33] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved Techniques for Training GANs. *ArXiv e-prints*, June 2016. 3

[34] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *ArXiv e-prints*, Sept. 2014. 4

[35] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, Jan. 2014. 5

[36] J. Sun, W. Cao, Z. Xu, and J. Ponce. Learning a Convolutional Neural Network for Non-uniform Motion Blur Removal. 2015. 3, 5, 7, 8

[37] R. Szeliski. *Computer Vision: Algorithms and Applications*. Springer-Verlag New York, Inc., New York, NY, USA, 1st edition, 2010. 2

[38] D. Ulyanov, A. Vedaldi, and V. S. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *CoRR*, abs/1607.08022, 2016. 5

[39] C. Villani. *Optimal Transport: Old and New*. Grundlehren der mathematischen Wissenschaften. Springer Berlin Heidelberg, 2008. 3

[40] O. Whyte, J. Sivic, A. Zisserman, and J. Ponce. Non-uniform deblurring for shaken images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010. 2, 8

[41] B. Xu, N. Wang, T. Chen, and M. Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015. 5

[42] L. Xu and J. Jia. Two-phase kernel estimation for robust motion deblurring. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2010. 2

[43] L. Xu, J. S. J. Ren, C. Liu, and J. Jia. Deep convolutional neural network for image deconvolution. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'14, pages 1790–1798, Cambridge, MA, USA, 2014. MIT Press. 5

[44] L. Xu, S. Zheng, and J. Jia. Unnatural L0 Sparse Representation for Natural Image Deblurring. 2013. 2, 7, 8

[45] R. A. Yeh, C. Chen, T. Lim, M. Hasegawa-Johnson, and M. N. Do. Semantic image inpainting with perceptual and contextual losses. *CoRR*, abs/1607.07539, 2016. 1

[46] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901, 2013. 4

[47] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*, 2017. 4