VEGAN: VISUAL ENHANCEMENT GENERATIVE ADVERSARIAL NETWORKS FOR DEBLURRING

Aaron Chong (tzehaoc), Adam Driscoll (jdriscol),
David Evans (dje1), David Robinson (davidr1) @cs.cmu.edu

ABSTRACT

The problem of blurry images has existed since the advent of cameras. Deblurring an image is directly useful for photography but is also useful for preprocessing inputs to deep learning networks. Classical deblurring techniques as well as various deep learning techniques have been used to deblur images in the past. We have used a Generative Adversarial Network to deblur images in the self-driving domain. We have followed the work of Kupyn et. al [9] but have improved their model by applying various loss function and data augmentation tricks. We have also applied three evaluation metrics different from Kupyn's to more accurately quantify the effect of deblurring.

1 INTRODUCTION

In recent years deep learning has made great strides in performing various complex computer vision tasks (e.g. classification, segmentation, etc.). However, the world is imperfect and image classification networks often have to classify noisy images. One form of noise is motion blur. Because the majority of these networks are trained and tested on carefully curated datasets which only contain clear, sharp images, the networks perform poorly on blurry images. If the blur in the images could be removed, then the networks could improve their accuracy.

Figure 1: Blurring Effect on Image Classification

A classical strategy to deblur images is to select a deblurring kernel and then convolve the kernel across the image to restore it. Selecting the deblurring kernel is often a difficult and imprecise task. The optimal deblurring kernel is unique to the motion blur in the image, and selecting the wrong kernel produces inaccurate results.

Kupyn et al. have recently presented DeblurGAN [9], which aims to solve this problem of motion blurred images through the use of a Generational Adversarial Network (GAN). In this project we will implement DeblurGAN and then adapt it to work with new datasets and validation methods.

2 RELATED WORK

We have established that to deblur an image, we first need to determine the blur kernel. Early work [11] focused on performing deblurring with a known blur kernel and using approaches such as the Lucy-Richardson algorithm to apply a deconvolution on the image using the blur kernel. The recent paradigm shift towards learning-based approaches has had significant impacts in computational

photography. Sun et al. [7] introduced a Convolution Neural Network (CNN) to estimate the blur kernel, and recent work with CNNs has explored deblurring approaches without a blur kernel.

Recently there has been research by Kupyn et al. in deblurring images with GANs [9]. GANs [4] are an architecture which define a generator and discriminator which compete against each other. The generator receives a noisy sample and attempts to recreate something valid, while a discriminator attempts to determine whether an input is real or not. GANs have large implications for this domain, as they are able to preserve textures and details within images and can generate images which are visually convincing.

3 Baseline Model

3.1 Network architecture

The originial implementation of DeblurGAN, uses a generator-critic model, inspired by the pix2pix architecture [5]. The generator resembles a conditional generative model with the condition being the blurred image and outputs a deblurred image. The generator's architecture, as seen in Figure 2, is composed of 2 strided convolution blocks with stride $\frac{1}{2}$, nine residual blocks, and two transposed convolution blocks. Each ResBlock consists of a convolution layer, instance normalization layer and ReLU activation, with 0.5 probability Dropout regularizations after the first convolution layer. The authors added global skip connections, to accelerate training and introduce further generalization to the model. The loss of the generator is the sum of the adversarial loss and a content loss. Kupyn et al. use the Wasserstein-GP loss as the adversarial loss and a perceptual loss for the content loss. The perceptual loss will be discussed with more detail in section 4.1.

$$\mathcal{L}_{gen} = \mathcal{L}_{adversarial} + \mathcal{L}_{content}$$

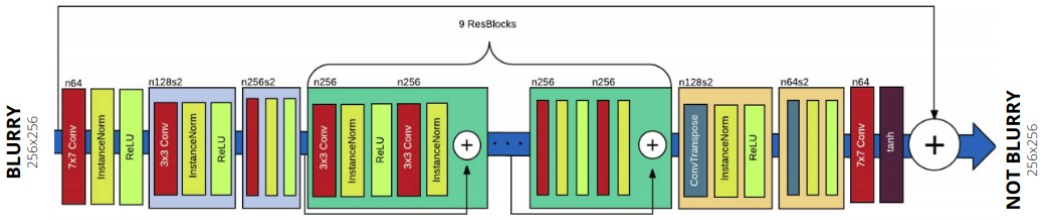

Figure 2: Conditional generator model used in the original paper, which takes in the blurry image and outputs a deblurred image.

The critic on the other hand is a simple fully convolutional network resembling PatchGAN [3], which receives images and attempt to classify them as ground truth or a deblurred image. The discriminator architecture can be seen in Figure 3. Each convolution layer is followed by InstanceNorm and LeakyReLU layers with leaking rate of 0.2 [9]. The loss function of the discriminator is the opposite of the adversarial loss of the generator and is also the Wasserstein-GP loss.

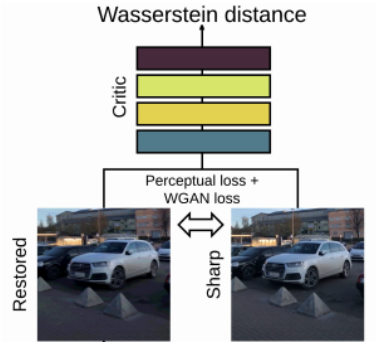

Figure 3: Critic model used in the original paper, which classifies whether images are sharp ground truths or deblurred images.

3.2 Shortcomings

The pretrained baseline model provided by the authors was evaluated and we noticed a few shortcomings in its results, namely in handling text as well as blurs in multiple scales and objects. Below we can see these problems on some of the test data for the pretrained model.

Figure 4: The top row shows the model failing to deblur the alphanumerical plate, while bottom row shows that the building in the background is deblurred well but the car remains very blurry.

4 Improvement Methods

In order to improve the baseline model deblurring performance on multi-scale deblurring, a failure case of the baseline highlighted in the previous section, we present the following modifications which are hypothesized to improve network performance.

4.1 Scaled Perceptual Loss

The original content loss that was implemented by the baseline used the mean-squared error between the feature maps of a pretrained VGG-19 network, specifically the fifth convolution layer. This is called perceptual loss. The idea stems from the idea that the generator should restore visual similarity of the image. In order to solve the problem of deblurring on multiple scales, we adapted the original perceptual loss to consider several layers of feature maps from the VGG-19 instead of only using a single feature map layer. This will encourage the network to reconstruct features across

multiple scales depending on the layer's receptive field and thus help the model restore a broader range of features across the image. Mathematically the scaled perceptual loss can be written as such.

$$\mathcal{L}_{spl} = \frac{1}{2}\sum_{k=1}^{K}(\phi_k(G(I_B)) - \phi_k(I_S))^2$$

The blurry image, I_B is passed through the generator G to be deblurred. This deblurred image is then run through a VGG-19 network and the k^{th} feature maps, ϕ_k are extracted from their respective layers of the VGG network. Our team extracted the feature maps from the second, fourth, eighth, and twelfth convolutional layers as can be seen in Figure 5. Each one of these layers are immediately before the max pooling layers in the VGG-19 and therefore the feature maps contain the most helpful features.

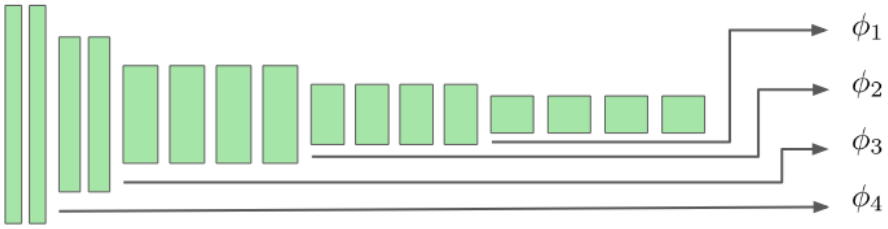

Figure 5: VGG-19 network and the convolutional layers feature maps were extracted from

The ground-truth image is also passed through the VGG network to extract ground-truth feature maps at the same layers. For each layer in the a feature map is extracted, the mean-squared error is calculated between the deblurred image's feature map and the corresponding ground-truth feature map. The losses across these layers are summed to form a final scaled perceptual loss that is used as the content loss for the generator.

4.2 Smoothness Loss

In addition to the scaled perceptual loss, we decided to add a smoothness loss to the content loss of the generator. Artifacts in the deblurred image can be caused by sharp gradients being added by the generator to try and minimize the scaled perceptual loss. We don't want these sharp gradients, unless there are sharp gradients in the ground-truth image. With this in mind, we draw on a loss function common in depth estimation networks.

$$\mathcal{L}_{sm} = |e^{-\nabla I_s}\nabla G(I_B)|$$

The loss tries to minimize the gradient of the deblurred image, but the loss is small when there is a strong gradient in the original ground truth image. This causes the deblurred image to try to smooth out the image but still replicate the gradients of the ground-truth image. The final content loss of the generator is finally calculated to be the sum of the smoothness loss and the perceptual loss or scaled perceptual loss, depending on which loss is selected.

$$\mathcal{L}_{content} = \mathcal{L}_{spl} + \mathcal{L}_{sm}$$

4.3 Random Scale Crop

The original authors augment the data by randomly cropping the input image into a 256 square. In order to modify the network to handle varying levels of blur across a wider variety of object scales, we propose a method to randomly select the crop size of the blurred and ground-truth imagse, before scaling it to the appropriate 256×256 input size for training. Our dataset was synthetically blurred using randomly generated trajectories, but with a fixed kernel size, therefore by randomly cropping and scaling, we are generalizing over a distribution of kernel sizes. One of the downsides of this method is that we are increasing the amount of noise in the training data when we up-sample it. We sample the random crop sizes from a normal distribution with a mean of 256 and a standard deviation of 50.

$$size \sim N(\mu = 256, \sigma = 100)$$

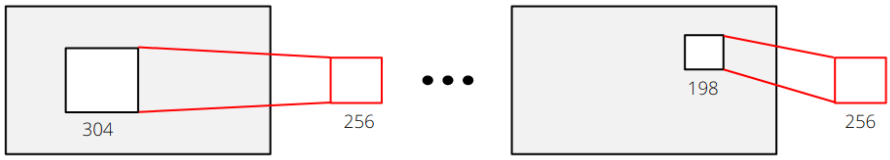

Figure 6: Random scaled cropping, with mean 256, standard deviation 50, with a target size of 256.

5 Dataset

Unfortunately, there are no large datasets available with blurred and unblurred image pairs. Faced with a similar problem, the authors of DeblurGAN used synthetic blurring techniques inspired by Boracchi and Foi [1]. Kupyn *et al*. blur all 1000 images of a GoPro dataset. Our team is interested in deblurring images in the self-driving domain, so we decided to use the same technique to blur the Mapillary Vistas dataset [8]. Mapillary Vistas is a collection of 25,000 street-level images, with different resolutions, aspect ratios and camera calibration parameters. The volume and variety of images were attractive in helping the team train a generator that was robust to various image scenes.

The synthetic blurring starts by designing a blur kernel. The blur kernel is a trajectory that represents the camera's motion that would cause the blur. We randomly generate the blur kernel by sampling from a set of continuous, random 2D positions through a Markov process which combines Gaussian perturbations and inertial considerations to create a realistic motion-blur trajectory. The trajectory is then converted into a blur kernel by using sub-pixel interpolation. Finally, the blur kernel is applied to the image by convolving it in 2D to produce the results in Figure 7.

Figure 7: Dataset Blur Results, **Left column:** Original image, **Middle column:** Synthetic blurred image, **Right column:** Blur kernel

6 Experiments

6.1 Evaluation Metrics

The authors of the original paper evaluated the performance of their network using a deep neural network for object detection, YOLO [6]. This allowed them to assess the deblurring results in terms of detection confidence, accuracy, recall, and F1 score. With much of the focus in computer vision, especially in the field of scene understanding, shifting away from bounding box detections, we propose to use other quantitative metrics that would give a better understanding of how a deblurred image compares against the ground truth sharp image. The task of quantitatively evaluating the performance of the deblurring network has presented us with a challenge in that, while it is very clear that the images are being restored through qualitative visual examination, many standard techniques for measuring image similarity do not tell the same story. The most naive measure of image similarity is the pixel-wise Mean Squared Error (MSE), or the related Peak Signal to Noise Ratio (PSNR), which are insufficient for measuring the restoration of structure within the image because of their

underlying assumption of pixel-wise independence. In fact, it is a well known case that blurred images, while having a large perceptual distance, have a relatively low MSE with the original image. The Structural similarity measure (SSIM) is another technique which, while approaching our desired metric of perceptual similarity, also fails to give meaningful results when applied to blurred images due to its failure to account for the nuances of human perception. As such, we must explore new options and ideas for evaluating the performance of our network.

6.1.1 Intersection Over Union

We propose evaluating our system using the intersection-over-union results of a pretrained deep network for pixel-wise semantic segmentation, specifically WideResnet38+DeepLabV3 which was implemented by the mapillary team when they introduced Inplace-BatchNorms [10].

Figure 8: Semantic segmentation for a Mapilliary image.

By performing pixel-wise segmentation on the deblurred image and the ground-truth image then calculating the interesection over union of the two images, we can calculate measure how well the generator deblurred the image. The objective function is show below,

$$\arg\min_{G} IOU(\theta_{seg}(I_S), \theta_{seg}(G(I_B)))$$

In order to calculate the intersection-over-union of the segmentation outputs, we generate a confusion matrix across all classes, which then the intersection and union can then be easily retrieved, shown below, with C being the confusion matrix.

$$IOU(P_1, P_2) = \frac{diag(C_{P_1,P_2})}{\sum_0^i C_{P_1,P_2}^{i,j} + \sum_0^j C_{P_1,P_2}^{i,j} - diag(C_{P_1,P_2})}$$

6.1.2 Perceptual Error

The visual deep learning community has discovered that features of the VGG network which have been trained on imagenet for a classification task can be used to construct a loss function which promotes perceptual similarity between a network output and its training label. In fact, we are training our network with a perceptual loss of this manner, and could use this loss as our similarity metric. However, our network is explicitly trying to minimize this quantity while training and therefore we need a more objective measure. [12] explores the ability of networks which are designed for specific visual tasks such as classification and recognition to undertake this task of measuring perceptual similarity.

Given two images that we want to measure the similarity of, we begin with a pass through the chosen visual similarity network (AlexNet or VGG) to compute the embeddings at each layer. Then, the activations are normalized in the channel dimension and scaled. The L2 distance is computed and the spacial average and layer average is taken to give one scalar measure of similarity shown in Figure 9.

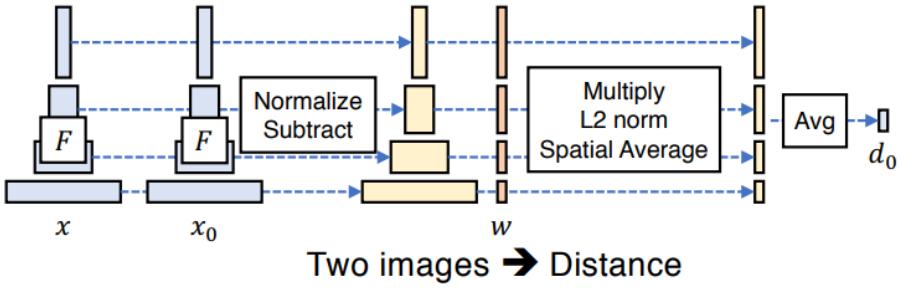

Figure 9: Diagram showing the basic concept implemented by [12].

6.1.3 Fourier Domain Error

The last error metric we decided to apply was Fourier domain error. As mentioned before, images and their blurred conjugates, while they look visually dissimilar, this difference is hard to quantify. The mean-square error between a blurred image and it's sharp pair is usually quite small because a blurred pixel is the average of the pixels around it, and therefore usually similar to the corresponding pixel in the sharp image.

In frequency space however, blurring an image has a significant effect because the blurring acts a low-pass filter that removes high-frequency components of the image. This effect can be seen by comparing the magnitude spectra of a blurred image and it's ground-truth pair in Figure 10. The intensity of the pixels in the spectra correspond to the energy at the corresponding frequency. The brighter the pixel, the more energy at the frequency. One can see blurring the image preserves many of the low frequency terms while the high frequency terms are attenuated.

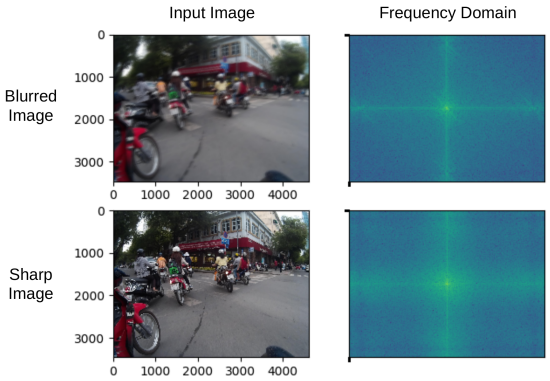

Figure 10: Blurry image compared to sharp image in frequency domain

The final Fourier domain error is calculated by finding the average mean-squared error between the magnitude spectra of the deblurred image and the ground-truth image.

6.2 RESULTS

We trained a total of five models to evaluate our methods: the baseline, three models each with a single one of our improvement methods, and one model with all three improvements methods. We trained the models with a NVIDIA GTX1080 TI GPU for a total of 15 epochs and 14,400 iterations per epoch. Our software was written to use Pytorch 0.4.0 and our implementation can be found here [2].

Each model was then tested on a set of 100 held out test images from our blurred Mapillary dataset. We measured our models with our three different metrics: IOU, perceptual similarity, Fourier domain error.

Figure 11 shows the visual results of our three-method combined model and the baseline model. Table 1 shows the quantified results of the baseline, each method, and the combined-method model.

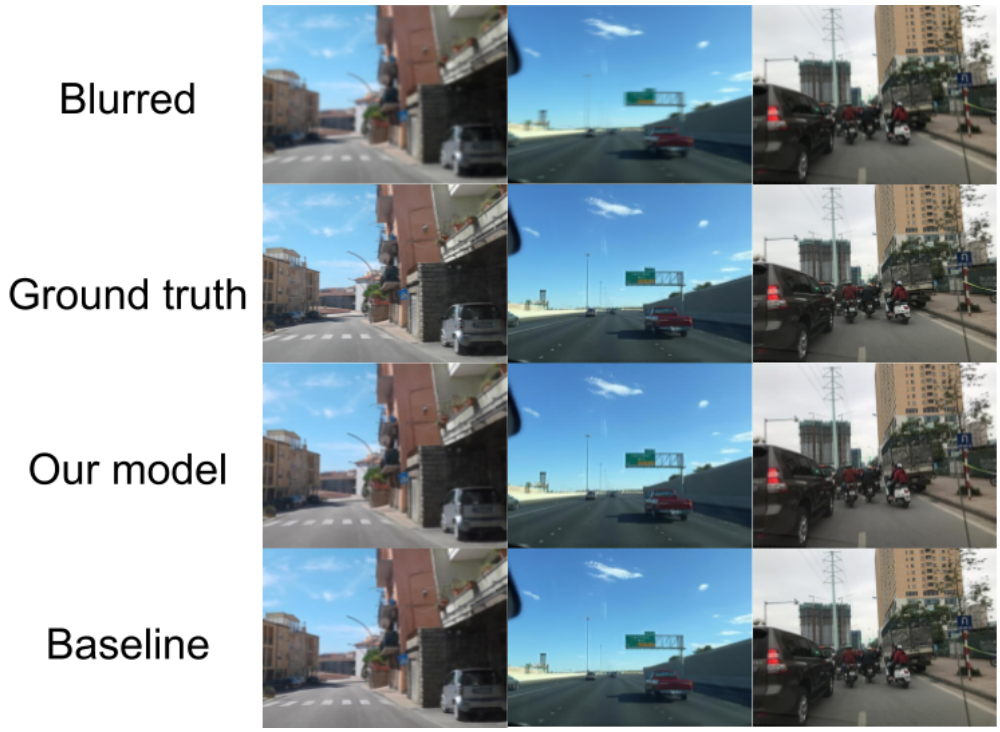

Figure 11: Visual results of deblurring

Model	Metric		
	IOU	Percep. Sim. Err.	Four. Dom. Err.
Blurred image	0.5140	0.2831	0.0804
Baseline	0.5270	0.2486	0.0632
Random scale crop	0.3206	0.4049	0.0594
Smoothness loss	0.5287	**0.2307**	**0.0584**
Scaled percep. loss	**0.5334**	0.2470	0.0634
All three methods	0.5276	0.2332	**0.0584**

Table 1: Quantified model performance

Our methods visually and quantitatively deblur the images. Each of our methods either meet or exceed the baseline performance across our three error metrics with the exception of the random scale crop on the IOU and perceptual similarity metrics.

The random scale crop technique when viewed visually seems to be producing outputs with good structural properties but a slight color hue offset. We hypothesize this is a bug on our end and may be a normalization issue. The color offset error would also explain why the Fourier domain error is still reasonable because in order to calculate the frequency spectrum, the channels are averaged, essentially making the image black and white.

Otherwise, we can state the combination of all three of our improvement methods were effective in improving the baseline. Specifically the smoothness loss made the largest impact.

It is difficult to visually compare the baseline and our methods. When directly comparing the two, the differences are almost impossible for a human to identify. Figure 12 compares a deblurred image by the baseline model and by our model. The two images look exactly the same. When comparing the pixel-wise difference between the two images, only slight differences in pixel values around the edge of objects are observed.

Figure 12: Baseline deblurred image compared to our model deblurred image

7 Conclusions

7.1 Discussions

The baseline model provided by original authors of DeblurGAN was evaluated, and after identifying its weaker aspects such as multi-scale deblurring, we came up with a number of methods to improve the results. We introduced a smoothness loss for training, in order to enforce smoothness on the deblurred output where gradients are smooth on ground truth images as well. A modification to the perceptual loss was also introduced, by comparing multiple layers of feature maps instead of just one of a trained VGG-19 network, we allow the model to generalize over features of multiple scales depending on the receptive field of that particular perceptual loss layer. Lastly, we augmented our training data, by implementing random scaled crops, making the model learn how to deblur over multiple scales of blur kernels. We trained a number of models using a combination of these techniques and have validated that using all three proposed methods together, achieved better results than the original paper's baseline implementation.

In order to evaluate the performance of the models, we also introduced a number of different evaluation methodologies. The first is the intersection-over-union score of the segmentation output of deblurred images versus that of the ground-truth images, which provides an idea of how salient the boundaries between different classes are. Secondly, we used a similar method to the perceptual loss, but following the implementation of [12], using the feature maps of multiple layers of AlexNet instead of VGG-19. Third, we introduced a similarity score by comparing the Fourier domain mapping of each image, this gives a good idea on both the sharpness and smoothness of the image. After testing, it is shown that evaluating over the Fourier domain produced consistent results, and shows that our best models performed better than the baseline implementation.

7.2 FUTURE WORK

Looking to next steps in the project, our team has several thoughts on how to improve our models even more. One shortcoming in our current model is how it smooths textures. This could be attributed to the smoothness loss. One solution to counter the corruption of textures is inspired by our Fourier domain error metric. We believe adding a loss function that tries to minimize the L2 distance in the frequency domain may be useful in recovering high frequency components from the deblurred image and thus restore more realistic texture.

Additionally, we could improve our method of synthetic blurring. In our current synthetic blurring method, the pixels are blurred affinely throughout the entire image along a trajectory. However, that is not an accurate model for camera motion blur. We could improve this blurring method by imposing camera projective geometry into the scene, segmenting the entire image into patches where a separate kernel be used to mimic true camera motion. A difficulty associated with this would be to obtain the camera intrinsics which took that particular photo. However, this could be easily obtained by merging multiple datasets using different cameras in order train the model to generalize over all types of cameras and motion.

REFERENCES

[1] G. Boracchi and A. Foi. Modeling the performance of image restoration from motion blur. In *Image Processing, IEEE Transactions*, August 2012.

[2] Tze Hao Chong, Adam Driscoll, David Evans, and David Robinson. VEGAN repository. `https://github.com/son-of-a-gan`, 2018. [Online].

[3] Ugur Demir and Gözde B. Ünal. Patch-based image inpainting with generative adversarial networks. *CoRR*, abs/1803.07422, 2018. URL `http://arxiv.org/abs/1803.07422`.

[4] M. Mirza B. Xu D. Warde-Farley S. Ozair I. Goodfellow, J. Pouget-Abadie. Generative adversarial nets. *In Advances in neural information processing system*, pages 2672–268.

[5] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *CVPR*, 2017.

[6] R. Girshick A. Farhadi J. Redmon, S. Divvala. You only look once: Unified, real-time object detection. In *ArXiv e-prints*, June 2015.

[7] Z. Xu J. Ponce J. Sun, W. Cao. Learning a convolutional neural network for non-uniform motion blur removal. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 769–777, June 2015.

[8] P. Kontschider. Mapillary vistas. URL `https://blog.mapillary.com/product/2017/05/03/mapillary-vistas-dataset.html`.

[9] M. Mykailych D. Mishkin J. Matas O. Kupyn, V. Budzan. Deblurgan: Blind motion deblurring using conditional adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2018.

[10] Samuel Rota Bulò, Lorenzo Porzi, and Peter Kontschieder. In-place activated batchnorm for memory-optimized training of dnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[11] R. Szeliski. Computer vision: algorithms and applications. In *Springer Science Business Media*.

[12] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.