Deep-Learning-based Image Denoising in Ophthalmology

Lukas Krenz

April 13, 2018

1 Introduction

Intraoperative medical imaging is an important part of modern surgery. During ophthalmic procedures, the surgeon manipulates anatomical structures of the retina with micron-scale maneuvers while observing the scene indirectly via microscope. The resulting magnification has the effect that only a small area is focused, other parts are either distorted or occluded. Additionally, the motion of the eye introduces blur. All these factors result in images which are corrupted by noise. This makes precise operations more difficult. Intraoperative images not only differ from the diagnostic ones in terms of quality. During surgery, instruments are present and the membrane is stained with a coloring agent to amplify its edges.

Our goal is to increase the resolution of retinal fundus images both for diagnostic and intraoperative images. Assuming that we have an image that was downsampled by an unknown operator, the goal is to find the best reconstruction of the original image. We consider two different cases:

- Downsampling by decreased spatial resolution. In our case we want to reconstruct an image that was downscaled by a factor of 2 or 4. This problem is called single image super resolution.
- Downsampling by decreased sharpness. For this case we try to increase the quality of images by removing blur.

An image restoration algorithm has to fulfill the following requirements to be considered useful in an intraoperative setting:

- All processing should happen in real time.
- It has to work with images of varying quality, level of zoom and different positions of surgical instruments.

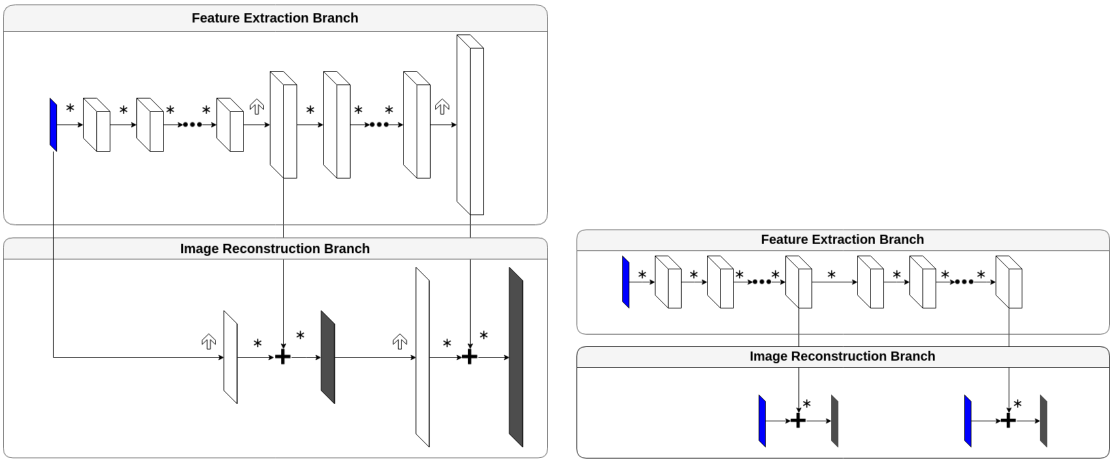
- The anatomical structure, the position of surgical instruments, and the color have to be conserved. Both blood vessels and the border of the membrane should be at least as clearly visible as in the original images. This implies that we have to preserve the image edges.

These constraints can be fulfilled by a deep learning approach.

We make the following contributions:

- We compare different loss functions, including adversarial training, and their suitability for the restoration of fundus images.
- We present deep-learning models that are able to reconstruct images corrupted by both mentioned downsampling operators in in real-time for small crop sizes.
- We evaluate the resulting algorithms with multiple metrics and discuss the consequences for intraoperative applications.

2 Architectures


Feature Extraction Branch
Image Reconstruction Branch
Feature Extraction Branch
Image Reconstruction Branch

(a) LapSRN [9]

(b) LapDeblur

Figure 1: Generator architectures. Blue and gray images show input and output images respectively. The symbol $(*)$ marks convolutions, $(\uparrow)$ resize-convolutions and $(+)$ corresponds to element-wise addition. All convolutions have 64 filters of size 3 with a padding and stride of 1. All convolutions except the ones in the resize-convolutions in the *image reconstruction branch* are followed by a leaky rectified linear unit with a negative slope of 0.2.

We use a modified **LapSRN** architecture (figure 1a) [9]. The network utilizes a Laplacian pyramid, it first performs 2x upscaling and then 4x upscaling.

This architecture consists of two branches. The first is the *image reconstruction branch* and upscales the low resolution image using simple upscaling filters. The second branch extracts features from the low resolution image using stacked convolutions and predicts the residual for each upscaled image of the first branch. We use a depth of 10 per output resolution for this *feature extraction branch.*

Our model differs in three ways from the original architecture:

- We use three-channel RGB images both as input and output instead of working only on the Y-channel.
- Instead of transposed convolutions we use resize-convolution blocks. They are composed of a nearest neighbor interpolation, a reflect-pad of size one and a standard convolution. They lead to fewer checkerboard artifacts than transposed convolutions, especially when they are used with adversarial learning [12].
- We only consider 4x upscaling here, but the approach can be easily extended to larger upscaling factors.

For the deblurring case, we modify the architecture (figure 1b) by removing the upscaling blocks from the *feature extraction branch.* Additionally, instead of using a *image reconstruction branch*, we compute the residuals for input images directly. Because all convolutions now operate in high-resolution space, we use 5 convolutions per output. This approach has the advantage that we can use similar models for both downsampling operators.

We use the **PatchGan** [6] discriminator for adversarial learning. This is a fully convolutional network that penalizes on a basis of 70×70 patches. Its architecture can be seen in figure 2. In contrast to [6] we use instance normalization [18] instead of batch normalization layers.

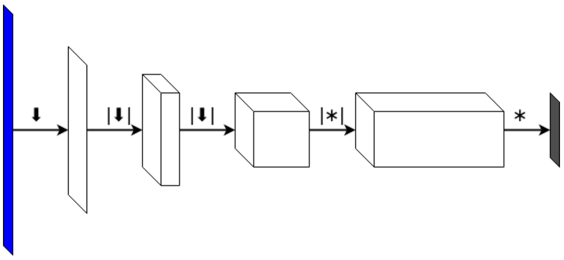
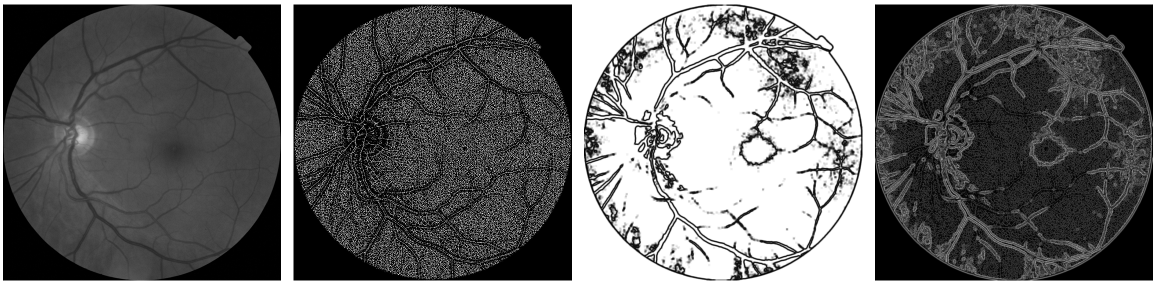


Figure 2: Discriminator architecture from [6]. The symbol $(|\cdots|)$ indicates instance normalization, $(\downarrow)$ and $(*)$ are convolutions with stride 2 and 1 respectively All convolutions have kernel size 4 and a padding of 1 and are followed by a leaky rectified linear unit with a negative slope of 0.2. The layers have an output size of 64, 128, 256, 512 and 1.

3 Loss functions

An appropriate choice of loss function is important for good reconstructions. In this work we combine three loss functions: A saliency weighted L_1-loss ensures the faithful reconstruction of important areas, a perceptual loss optimizes visual similarity and an adversarial loss reconstructs high-frequency image components. This ensemble of loss functions is inspired by [11] and is optimized for the reconstruction of retinal images.



(a) Input, Grayscale (b) I_c (c) $1 - I_e$ (d) I_{sal}

Figure 3: Components of a saliency map. Brighter pixels indicate a higher weight.

While the images in standard super-resolution tasks do not share a lot of common structure, retinal fundus images are visually similar to each other. This can be used to create hand-crafted saliency maps that highlight relevant pixels. We use maps that are similar to the ones of [11]. These saliency maps consists of two components: curvature and local entropy. The map and its components are shown in figure 3.

The curvature map highlights fine structure in the image and can be computed as

$$I_c = \frac{\boldsymbol{f}_{xx}\boldsymbol{f}_y^2 + \boldsymbol{f}_{yy}\boldsymbol{f}_x^2 - 2\boldsymbol{f}_x\boldsymbol{f}_{xy}\boldsymbol{f}_y}{(\boldsymbol{f}_x^2 + \boldsymbol{f}_y^2)^{1.5}}, \tag{1}$$

where $\boldsymbol{f}$ is the intensity of the image and subscripts represent derivatives [11]. We approximate the image derivatives by convolving the image with derivatives of Gaussians with $\sigma = 1$. Undefined pixels (due to division by zero) of the feature map are set to zero. We clip the resulting feature map to the range of $[0, 1]$.

The local entropy map highlights pixels that are in a neighborhood that contains more information. We compute the entropy for each pixel s by

$$I_e = -\sum_{s_i \in P_s} p(s_i)\log(p(s_i)), \tag{2}$$

where p is the probability that a pixel s has an intensity of s_i in a patch P_s [11]. The probability is estimated by a normalized histogram with eight bins of equal size, computed for each patch separately. A patch is composed of the pixel itself and its surrounding neighborhood of size 7×7. We then convolve the image with a Gaussian filter with $\sigma = 0.5$ to remove high-frequency noise. Finally, we normalize I_e to the range $[0, 1]$ and use $1 - I_e$ for our saliency map. This map highlights compact regions of the image.

We then compute the uniqueness of each pixel of a feature map f with

$$U(m) = \sum_{o \in P_c} w(c, o)|m(c) - m(o)|, \tag{3}$$

where c corresponds to the pixel in the middle of a patch P_c of size 7×7. The function $w(a,b) = \exp\left(d(a,b)\right)$ weights the pixels by their Euclidean distance d. All uniqueness maps are normalized to the range $[0,1]$.

Finally, we combine the uniqueness maps of both components linearly

$$I_{\text{sal}} = 0.4 \cdot U(I_c) + 0.6 \cdot U(1 - I_e) \tag{4}$$

to obtain the saliency map for each image [11].

We then use this map to weight a pixel-wise error, in our case the Charbonnier loss. It is a differentiable relaxation of the L_1 loss which leads to sharper edges compared to a standard MSE-loss [9]. Our weighted loss can be described by

$$L_{\text{sal}}(\hat{\boldsymbol{f}}, \boldsymbol{f}) = \|I_{\text{sal}}(\boldsymbol{f}) \circ \sqrt{(\hat{\boldsymbol{f}} - \boldsymbol{f})^2 + \varepsilon}\|_1, \tag{5}$$

where ε is a parameter set to $1e-6$ and $\circ$ denotes the element-wise product. The ground truth image is represented by $\boldsymbol{f}$ and the reconstruction by $\hat{\boldsymbol{f}}$.

The second component is the perceptual loss [7]. It does not compare images pixel-wise but considers the difference of filtered images. We use the feature activations of the first and second pooling layer of a VGG-16 network [16] pre-trained on Imagenet. The resulting loss is

$$L_p(\hat{\boldsymbol{f}}, \boldsymbol{f}) = \sum_{l \in \{\text{pool}_1, \text{pool}_2\}} \|\phi_l(\hat{\boldsymbol{f}}) - \phi_l(\boldsymbol{f})\|_2^2, \tag{6}$$

where ϕ_l denotes the activations of the layer l. This leads to perceptually better images but increased MSE.

We use an adversarial loss as our third and final building block, similar to [10]. Training then resembles a two-player game between a generator G and a discriminator D. The discriminator tries to decide whether an image is a ground truth high-resolution image or an low resolution image that was upsampled by the generator. The generator is trained to fool the discriminator. Both networks are trained alternatingly. This is realized as the optimization problem [5]

$$\min_G \max_D \mathbb{E}_{\boldsymbol{x} \sim P_r}\left[\log(D(\boldsymbol{x}))\right] + \mathbb{E}_{\hat{\boldsymbol{x}} \sim P_g}\left[\log(1 - D(\hat{\boldsymbol{x}}))\right]. \tag{7}$$

To improve stability, the generator minimizes

$$L_a = -\mathbb{E}_{\hat{\boldsymbol{x}} \sim P_g}\left[\log(D(\hat{\boldsymbol{x}}))\right] \tag{8}$$

instead. The patch-discriminator does not predict a single probability per image but rather one per image patch. Thus, the probability distributions P_r and P_g correspond to the distribution of real high-resolution patches and super-resolved patches respectively. This discriminator design is able to discover high-frequency image details while relying on the other two losses for low-frequency content [6]. Note that we only penalize the last output image by the adversarial loss.

The total loss function is then

$$L_{\text{total}}(\hat{\boldsymbol{f}}, \boldsymbol{f}) = \frac{1}{3W_f H_f} \left(\sum_{\boldsymbol{f}, \hat{\boldsymbol{f}}} 5L_{\text{sal}}(\hat{\boldsymbol{f}}, \boldsymbol{f}) + 0.12L_p(\hat{\boldsymbol{f}}, \boldsymbol{f}) \right) + 0.01L_a, \tag{9}$$

where the sum runs over all model outputs $\hat{\boldsymbol{f}}$ and their corresponding ground truth $\boldsymbol{f}$. Losses are normalized by number of channels (3), image height H_f and width W_f. The weights for saliency and perceptual loss are chosen such that they are of similar size, the adversarial weight is chosen empirically.

4 Implementation & Training Details

We use the Messidor dataset [2] which consists of 1200 high resolution fundus images. The black borders are removed. We use 80% of the dataset for training, the rest is used for validation. The high resolution images are augmented by:

- A random scaling with a factor uniformly distributed between 0.5 and 1.0. We rescale using bicubic downsampling.
- Random crop of size 128 × 128 chosen by rejection sampling: If a crop contains more than 50% black pixels (gray-scale value smaller than 90) or no vessels, it is rejected and another crop is chosen. The vessels are detected using a pre-computed Frangi filter [3]. We consider a crop to contain no vessels when fewer than 64 pixels are marked as vessels by the Frangi filter. This threshold was chosen empirically.
- Random rotation by either $0, 90, 180,$ or 270 degrees.
- Random vertical and horizontal flip, each with a probability of 50%.
- Specular reflections with a probability of 25%. They are simulated by increasing the intensity of the image in a circular mask (post-processed with a Gaussian filter) by a random intensity. The specular reflection mirror the usage of light sources during surgery.

Note that all augmentations (except specular reflections) are also applied to the pre-computed saliency maps and the random scaling is applied on the vessel segmentations as well. The scaling, rotation and flipping augmentations are also used by [9].

We obtain low resolution images by a Gaussian blur with maximum radius of three for denoising. For the super-resolution network, we first apply a Gaussian blur with a maximum radius of two and then use bicubic downsampling with factors two and four. The blurring is done by first selecting a random total blur radius, sampled uniformly between zero and the maximum blur. We then distribute the blur such that the first input image is blurred with half strength. This blur radius can be computed by $\left(\text{radius}_{\text{total}}/\sqrt{2}\right)$, using the fact that a convolution of two Gaussians is a Gaussian.

Finally, images and saliency maps are converted to tensors of range $[0, 1]$. Images are scaled by subtracting $\begin{bmatrix}0.485 & 0.456 & 0.406\end{bmatrix}$ and dividing by $\begin{bmatrix}0.229 & 0.224 & 0.225\end{bmatrix}$. This is the normalization that is expected from the pre-trained VGG-16 network.

We use a batch size of 64 for all experiments. An epoch thus consists of 15 gradient updates.

All weights of the generator are initialized by Xavier initialization [4], all weights of the discriminator are drawn from a normal distribution with mean zero and variance of 0.02 [6]. We first train solely the generator without adversarial loss. It is optimized by ADAM [8] with an initial learning rate of 10^{-4} for the super-resolution network and 10^{-5} for the deblurring network. We use a weight decay of 10^{-5}. The super-resolution generator is trained for 9999 epochs, the deblurring generator for 6666 epochs. The learning rate is divided by ten every 3333 epochs.

The resulting network is then used to initialize the adversarial training, which continues for another 6666 epochs. We use the same learning rate for both networks, starting with an initial value of 10^{-5}, using the same learning rate schedule as for the initial training. Both networks are optimized by ADAM with the momentum term β_1 set to 0.5 for improved stability [14]. No weight decay is used in this stage.

The network was implemented in Pytorch [13] and was trained on a Titan X. Training took 20 h for the super-resolution network without adversarial loss and an additional 29 h for the adversarial training. The deblurring network took 34 h for the initial training and an additional 44 h for adversarial training.

5 Evaluation

The evaluation of image restoration algorithms is difficult because most metrics do not correlate with human perception. This is why we use multiple metrics, some of them tailored for retinal image restoration.

To ensure a fair comparison, images are first reflect padded to have a size divisible by 4 for super-resolution and 16 for deblurring. This padding is then removed before all comparisons. We cut off an additional stripe of size equal to the upsampling factor for the super-resolution network.

We use the MSE-based metric PSNR to compare images on a pixel-wise basis. Additionally, we use the structural similarity (SSIM) [20] which uses a model based on human perception. Following standard procedure, we apply these metrics on gray-scale versions of our images. For our chosen application, the correct reconstruction of image gradients is important, for example to enhance the border of the retinal membrane. To verify this, we compute the gradient magnitude with a Sobel filter and compare the reconstruction.

Similarly to [11] we use vessel segmentation to evaluate the accurate reconstruction of image details. For this we use two methods:

1. The Frangi filter [3] is a simple segmentation method. We use the implementation of [19] with parameters $\beta_1 = 0.7, \beta_2 = 0.01$ and a scale range of $[0, 3]$. Pixels with a intensity of 0.2 or larger are marked as vessels. These parameters were found with a grid search on the training set. The method is not robust, sometimes a

blurry images leads to a more accurate segmentation. This is why we compare the reconstruction of the Frangi segmentation instead of the segmentation accuracy.

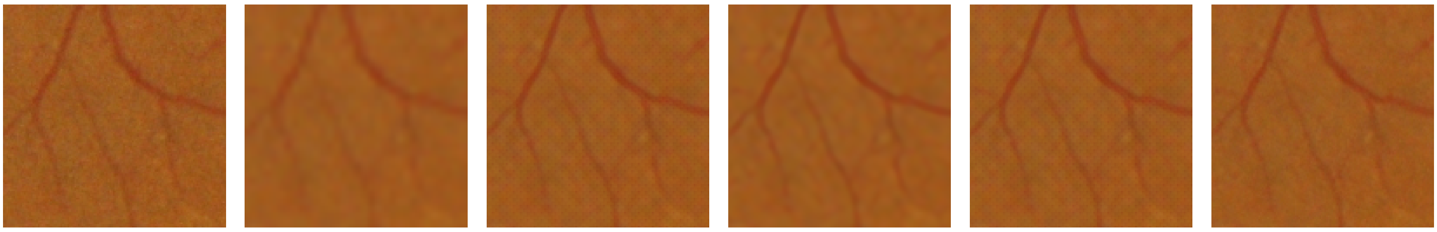
2. As an example for a state-of-the art deep-learning based algorithm we use the **Retina UNet** [1]. It is based on the **UNet**-architecture [15]. For this method we report the area under the ROC-curve (AUC).

We evaluate this on the testing set of the Drive dataset [17], for which a ground truth segmentation is available.

We compare the super resolution network with bicubic interpolation.

Table 1: Results for super resolution models on our validation dataset (Messidor [2]) for both possible upsizing factors. The full model is not compared for the 2× model because the adversarial loss is only applied to the largest output image. Best results are bold.

Model	Factor	PSNR	SSIM	Sobel-MSE 1×10^4
Bicubic	2	44.67	0.972	2.56
Perceptual	2	42.21	0.945	1.25
Saliency	2	**45.30**	**0.973**	1.33
Saliency + Perceptual	2	44.83	0.969	**1.22**
Bicubic	4	40.69	0.951	8.35
Perceptual	4	42.22	0.943	2.53
Saliency	4	**42.82**	**0.953**	3.59
Saliency + Perceptual	4	42.23	0.945	**2.51**
Full	4	41.78	0.942	2.64



(a) HR (b) Bicubic (c) Perceptual (d) Saliency (e) Sal. + Perc. (f) Full Model

Figure 4: Upscaling results for an 128^2 patch from our validation set (Messidor [2]).

We first evaluate on our hold-out validation dataset (table 1). For an example image patch for this dataset, see figure 4. We also evaluate on the Drive [17] dataset (table 2). An example segmentation for all loss function is shown in figure 5. To evaluate the effectiveness of our chosen loss function, we trained different combinations with the same training schedule:

Table 2: Results for super resolution models on the Drive [17] (testing) dataset. AUC corresponds to area under the ROC curve achieved by running the retina-unet [1] on the upscaled images. Best results are bold.

Model	PSNR	SSIM	Sobel-MSE 1×10^4	Frangi Reconstruction Acc.	AUC UNet
Ground Truth	∞	1.00	0.00	1.00	0.979
Bicubic	35.27	0.92	29.87	97.05	0.852
Perceptual	38.90	0.93	8.61	**98.62**	0.943
Saliency	**39.50**	**0.94**	8.55	98.35	0.921
Saliency + Perceptual	39.20	0.93	**7.84**	**98.62**	0.946
Full	38.85	0.93	8.29	98.51	**0.948**

Saliency The saliency weighted L_1-loss leads to an accurate, albeit blurry, reconstruction of important features of the anatomy. It results in the highest PSNR and SSIM. Some textural details are missing and edges of vessels are not clear. It has the worst segmentation results, tiny details are lost by this reconstruction.

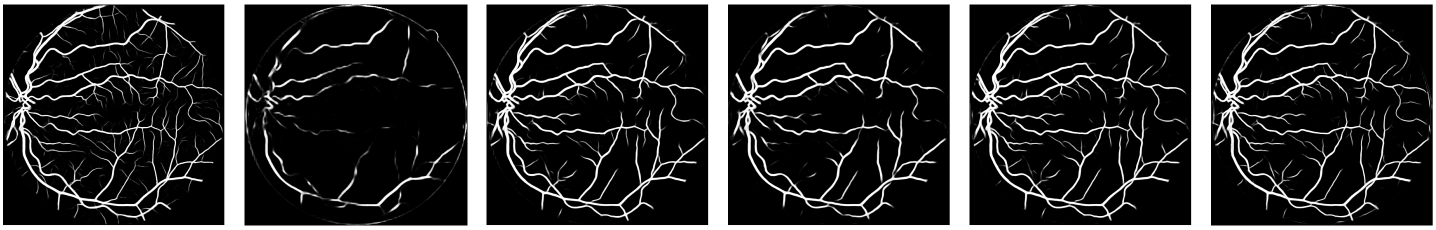
Perceptual The perceptual loss results in lower PSNR and PSNR but the images are more realistic and visually more pleasing. It introduces regular artifacts and leads to the best reconstruction of the Frangi segmentation. This loss achieves a **UNet** AUC that is nearly as good as the loss ensemble result, indicating that it is the most important component for an accurate segmentation.

Saliency & Perceptual Combining both losses results in a reconstruction that is a good compromise between faithful pixel-wise reconstruction and perceptual quality. The artifacts are still visible but with a smaller intensity. It has the lowest Sobel-MSE which indicates that it is able to restore image gradients more correctly. It restores the Frangi segmentation as correctly as the perceptual loss alone and has the second best **UNet** segmentation.

Saliency & Perceptual & Adversarial The addition of the adversarial loss improves the reconstruction of texture, the images are sharper. Additionally, it removes the artifacts introduced by the perceptual loss. It leads to the best AUC for the **UNet** segmentation. The downside of this loss is that the adversarial component is not trained to reconstruct the image correctly. This can lead to additional failure cases where the network hallucinates details, such as additional blood vessels.

Overall, our loss ensemble results in both correct reconstruction and visually convincing results. This is true as well for the $2\times$ scaled output. The adversarial loss trades faithful reconstruction for sharper images. With and without it, our proposed loss combination achieves competitive results on both datasets.

The deblurring network is compared to a standard **UNet** architecture [15] with MSE loss, trained in the same manner as our network. We use an implementation that does not crop feature maps for the skip connections but rather pads them such that they have



(a) HR (b) Bicubic (c) Perceptual (d) Saliency (e) Sal. + Perc. (f) Full Model

Figure 5: Segmentation results on Drive [17] testing set.

the same shape. This is a common architecture for bio-medical applications and similar networks are used successfully as generators for image-translation tasks [6]. Our network is able to improve the perceived image quality (figure 7), SSIM and segmentation AUC (figure 6). Note that the **UNet** achieves a better SSIM-value but worse AUC.

Even though we did not optimize our implementation for speed, super-resolving image patches of size 128^2 and deblurring patches of size 256^2 is possible in real-time (figure 8).

A preliminary evaluation on intraoperative data did not show a significant improvement for both deblurring and super-resolution. This could be caused by poor video quality.

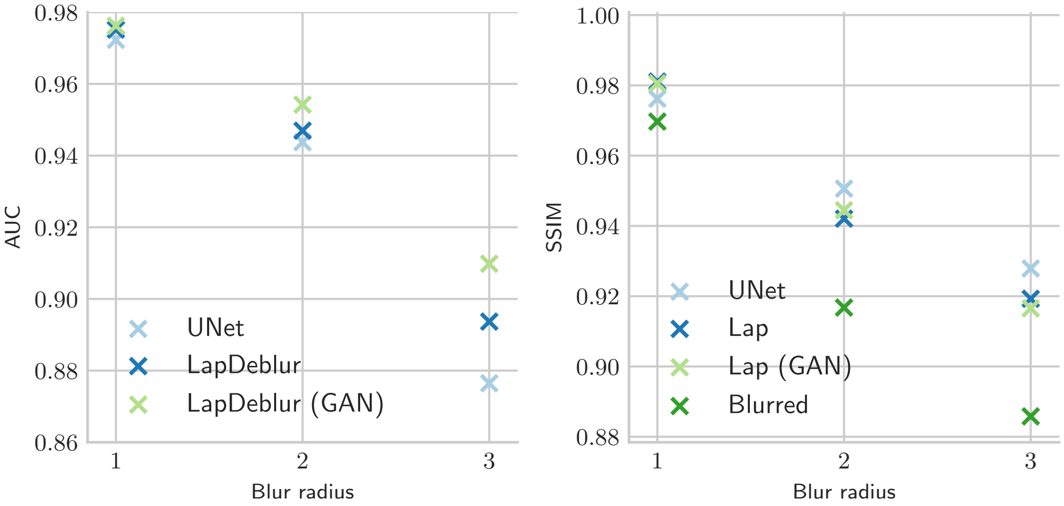
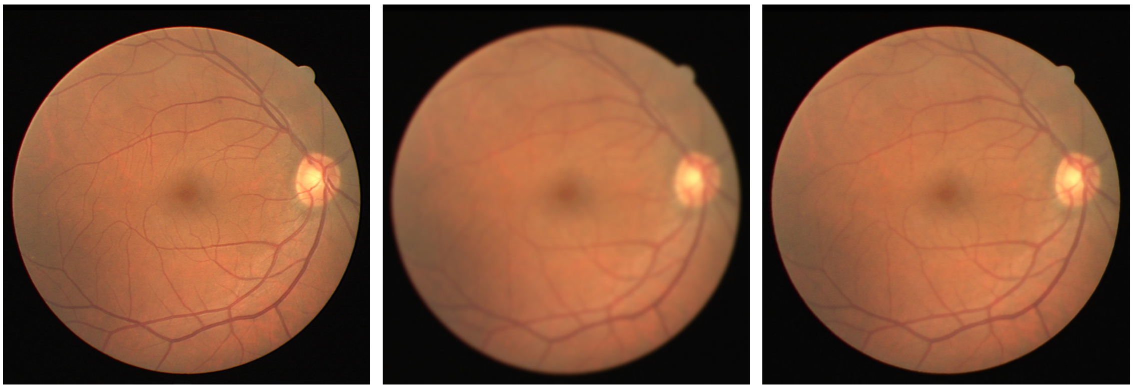
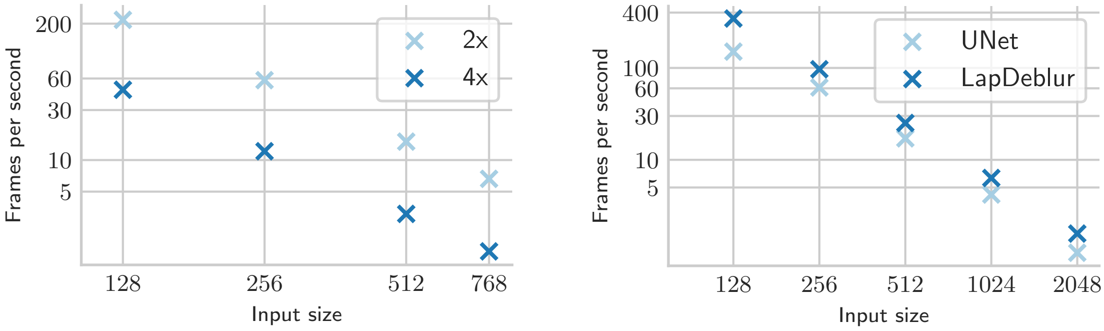


Figure 6: Performance metrics for deblurring on the Drive [17] dataset (testing). Segmentation on blurred images directly leads to an AUC of 0.96, 0.83 and 0.65 respectively.



(a) GT (b) Blurred, Radius 3 (c) Deblurred with full model

Figure 7: Example for deblurring. Image from Drive (test) [17]



(a) Speed for upscaling (b) Speed for denoising

Figure 8: Frames per second vs. input image size. Measured on a Titan X.

6 Summary

Our proposed models work well for diagnostic images. They are efficient and result in both faithful and perceptually convincing images. While the individual loss functions result in good reconstructions, our chosen ensemble is clearly superior.

The algorithms satisfy all our constraints for intraoperative usage:

- They work in real-time for small patch sizes without performance optimization. If a faster speed is desired our super-resolution network is also able to output images upscaled by a factor of two.

- The deblurring network is robust to different levels of blur. It is able to improve the segmentation results on all tested blur intensities.

  The super-resolution network is robust to changes in blur and image resolution due to the used data augmentations.

- All networks except the ones trained with adversarial loss retain the image content. Our evaluation shows that the proposed loss ensemble recovers vessels, image gradients and overall detail better than all single loss functions. It retains the anatomical structure including fine details.

We have thus shown that our proposed methods can be used successfully as a pre-processor for other computer vision tasks or to provide an improved digital zoom.

References

[1] Orobix Srl Daniele Cortinovis. *retina-unet*. 2018. URL: `https://github.com/orobix/retina-unet` (visited on 03/08/2017).

[2] Etienne Decencière, Xiwei Zhang, Guy Cazuguel, Bruno Lay, Béatrice Cochener, Caroline Trone, Philippe Gain, Richard Ordonez, Pascale Massin, Ali Erginay, et al. "Feedback on a publicly distributed image database: the Messidor database." In: *Image Analysis & Stereology* 33.3 (2014), pp. 231–234.

[3] Alejandro F Frangi, Wiro J Niessen, Koen L Vincken, and Max A Viergever. "Multiscale vessel enhancement filtering." In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 1998, pp. 130–137.

[4] Xavier Glorot and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks." In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. 2010, pp. 249–256.

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

[6] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. "Image-to-Image Translation with Conditional Adversarial Networks." In: *CVPR* (2017).

[7] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. "Perceptual losses for real-time style transfer and super-resolution." In: *European Conference on Computer Vision*. Springer. 2016, pp. 694–711.

[8] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization." In: *arXiv preprint arXiv:1412.6980* (2014).

[9] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. "Deep Laplacian Pyramid Networks for Fast and Accurate Super-Resolution." In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2017.

[10] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew P. Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 105–114.

[11] Dwarikanath Mahapatra, Behzad Bozorgtabar, Sajini Hewavitharanage, and Rahil Garnavi. "Image Super Resolution Using Generative Adversarial Networks and Local Saliency Maps for Retinal Image Analysis." In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2017, pp. 382–390.

[12] Augustus Odena, Vincent Dumoulin, and Chris Olah. "Deconvolution and Checkerboard Artifacts." In: *Distill* (2016). DOI: `10.23915/distill.00003`. URL: `http://distill.pub/2016/deconv-checkerboard`.

[13] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. "Automatic differentiation in PyTorch." In: *NIPS-W*. 2017.

[14] Alec Radford, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." In: *arXiv preprint arXiv:1511.06434* (2015).

[15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241.

[16] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." In: *arXiv preprint arXiv:1409.1556* (2014).

[17] Joes Staal, Michael D Abràmoff, Meindert Niemeijer, Max A Viergever, and Bram Van Ginneken. "Ridge-based vessel segmentation in color images of the retina." In: *IEEE transactions on medical imaging* 23.4 (2004), pp. 501–509.

[18] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. "Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis." In: *Proc. CVPR*. 2017.

[19] Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu, and the scikit-image contributors. "scikit-image: image processing in Python." In: *PeerJ* 2 (June 2014), e453. ISSN: 2167-8359. DOI: `10.7717/peerj.453`. URL: `http://dx.doi.org/10.7717/peerj.453`.

[20] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. "Image quality assessment: from error visibility to structural similarity." In: *IEEE transactions on image processing* 13.4 (2004), pp. 600–612.