Joint Deep Demosaicing and Denoising For Interleaved X-Ray Channels

Jan Emrich
mail@janemrich.de

Frederik Wegner
wegnerfrederik@gmail.com

Faraz Saeedan
saeedan.faraz@visinf.informatik.tu-darmstadt.de

Abstract

Using deep neural networks for denoising and demosaicing of raw RGB and conventional X-Ray images has already given good results. This lab report documents how to transfer deep learning for demosaicing of RGB images and Noise2Self denoising to X-Ray images with interleaved channels. We note challenges of N2S training and suggest improvements. Further, we investigate how training demosaicing and denoising jointly can improve model performance as well as model runtime during inference.

1. Introduction

X-Ray Baggage Detectors need to trade off resolution, noise and speed. To improve resolution and noise level while maintaining speed, Smith developed a new detector. It still has the resolution limitation in raw data output, but enables the opportunity to improve resolution through demosaicing.

As the machine still needs to trade off noise with speed, raw output is very noisy. Noise can not be easily reduced through clean target training, because it is very hard or even impossible to obtain such data due to the way the machine works. Fortunately, there were new methods for self-supervised denoising introduced recently.

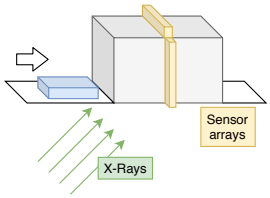


Figure 1: Two sensor lines capture a top and side view simultaneously.

Therefore we learn deep neural networks for the demosaicing and denoising tasks. Furthermore we join these tasks in a single model to improve model performance and runtime during inference.

Smith Detectors Smith produces multi-channel X-Ray detectors of different sizes and layout. These detectors are specifically designed and used for scanning baggage for unwanted and potentially harmful contents, such as weapons and explosives. The sensors are arranged in a line forming a sensor array. A detector (fig. 1) has two sensor arrays that capture a top and a side view. On a conveyor belt, the object that is scanned is moved across the sensor arrays while X-Rays are cast from the opposite site of the conveyor through the object onto the sensors. Each measurement is a row in the output picture. Varying belt speed and casting frequency directly affects the resolution or pixel size of the image along the height-/y-/column-axis as well as the noise in the image.

Sensor array layout A sensor array has two types of sensors, a high and a low frequency sensor, which combine into a two channel image. These sensors can be placed behind each other because they measure different rays and only interfere with the rays they capture (fig. 2). When combined the measurements of two channels at one location can be used to infer the material the rays had passed through. This information is then visualized with a color code as an RGB image as shown in fig. 3. The colorization algorithm was developed by Smith and is closed source but we were granted access to a binary executable.

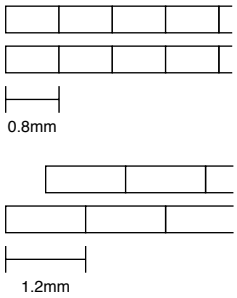


Figure 2: Top: PRO sensor arrays, bottom: SHARP sensor arrays. Pixels of the SHARP sensor are wider and the high channel is shifted to the right by half a pixel.

Detector types Smith fabricates sensors of different sizes. Smaller sensors result in higher resolution images along the x-/column-/width-direction of the image while larger sensors reduce the noise in the image. We work with images from 0.8mm and 1.2mm sized sensors. Two build configu-

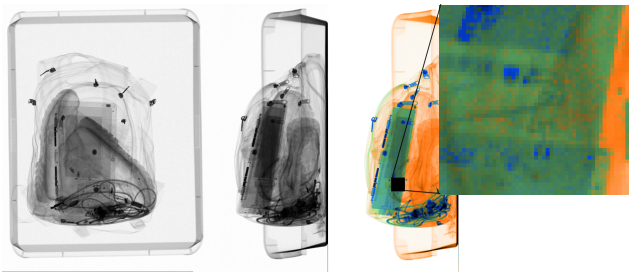


Figure 3: Left and middle: high channel image of the same object. Right: A color coding calculated from both high and low images. We only see the high image the low channel image is not displayed. An enlarged patch of the colorized image makes the effect of noise visible.

rations by Smith called *PRO* and *SHARP* are relevant (see fig. 2) to us. Due to the shifted sensor arrays, the SHARP detector output can be regarded as a mosaiced image. Despite having less sensors than the PRO detector, the SHARP detector can produce higher quality images because it takes less noisy measurements and the shift in the sensor array can be exploited through demosaicing. Less noisy measurements are preferable because they allow a higher scanning frequency and therefore faster belt speed.

2. Related Work

2.1. Demosaicing

Deep CNN have already been successfully trained to demosaic RGB images [5, 4]. This is approached by using a dataset of demosaiced ground-truth images and artificially mosaicing them by applying a bayer pattern filter. The network is then trained to recunstruct the original image from the artificially mosaiced one. The related works we present generally differ in whether they learn a residual on the input or not. Also different model architectures are used. Residual learning a model m to predict y from x means calculating the loss l as $l = lossfunction(m(x) + x, y)$ A residual network (short ResNet) is not a residually trained network(fig. 18). But residual learning is not required as a ResNet can jointly demosaicing and super-resolut RGB images [8]. The input is convoluted into a 256 dimensional feature tensor that is then passed through 24 residual blocks. The last convolution upsamples 256 channels to a 3 channel RGB output of higher resolution. Another architecture called ResNet Bottleneck achieves better peak signal to noise ratios on sRGB datasets [6]. The Bottleneck ResNet also upsamples the input to 256 channels and then passes it through 10 residual blocks to finally downsample it to a 3 channel RGB output (fig. 18). We reimplement the Bottleneck ResNet and use it as a baseline to compare agains.

2.2. Denoising

In this section we briefly discuss learning-based methods for image denoising.

Clean Target With available clean target data, UNet architectures can be trained to very accurately denoise images[7].

Self-supervised Recently, a new learning-based approach to denoising called Noise2Noise was introduced by Lehtinen *et al.* [3]. They found that predicting one noisy signal $\hat{x_1} = x + n_1$ from another statistically independent noisy signal $\hat{x_2} = x + n_2$ (with the same underlying signal x) will result in the true signal x. This works with all noise that has a zero-mean when using a L_2 loss, or similarly with zero median when using L_1 loss. Assuming zero-mean noise and that we want to optimize the prediction using a CNN with parameter mappings f_θ and loss L we have

$$\arg\max_\theta \sum L(f_\theta(\hat{x_1}), \hat{x_1}). \tag{1}$$

With $L = L_2(x, y) = (x - y)^2$, the minimum will be the expectation of the signal:

$$x = \mathbb{E}\{\hat{x}\}. \tag{2}$$

As the expectation of the noise is zero, the minimum will be the true signal. Therefore Noise2Noise can be used to train denoising from noisy image pairs.

Single self-supervised Based on the idea of Noise2Noise, two groups found another approach to denoise images only requiring single noisy image samples. They called it Noise2Void [2] and Noise2Self [1]. This approach requires to assume that the signal is pixel-wise dependent and the noise is pixel-wise independent. With this assumption, training as in Noise2Noise with single noisy images and a slight modification can be used to denoise: Use the same noisy image for the input and the output, but mask pixel in the input and predict their value. Then only use the loss from these predicted pixel to train. As the noise is pixel-wise independent, only its *a priori* expectation can be predicted, while the signal can be estimated from its surrounding pixel.

3. Methodology

3.1. Datasets

We have datasets of both the PRO and the SHARP detector. The same set of objects was passed through both detectors. Because a detector has two sensor lines (fig. 1) we get two images from the PRO and two images from the SHARP detector for each object. Figure 3 shows images

from the PRO detector. There are 99 objects each imaged from the top and the side, yielding 198 images. We split off 10 percent of the data for final evaluation. During training we use 10 percent of the training data for validation. This means our models are trained on 81 percent of the data.

As the images have different sizes, we extract equally sized patches (128 x 128) at random locations from each image to be able to form batches during training. The number of patches per image is usually set to 6 or 8. Sensor measurements range from 0 to 65535 and have the maximum value if the rays were unobstructed. We transformed the values into [0,1] range and invert them such that the background is zero and object have a value. To make learning more efficient the images are cropped to remove unneeded background before extracting the patches.

Simulated SHARP from PRO Our goal is to enhance the mosaiced images taken with the shifted 1.2mm sensor of the SHARP machine. As there is no demosaiced ground-truth we cannot use the SHARP images for training, but we have higher resolution demosaiced images from the 0.8mm sensor of the PRO machine. These are used to simulate mosaiced images of an imaginary 1.6mm SHARP sensor for which the corresponding PRO images serve as a ground truth. This is done by averaging adjacent 0.8mm pixels of the PRO images. (See fig. 4).

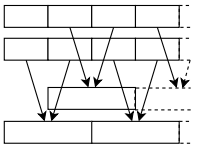


Figure 4: Top: PRO data, Bottom: simulated SHARP data

SHARP raw The images from the SHARP sensor that we want to improve. We apply our demosaicing model to these images during evaluation. We also train to denoise these images using the noise to self approach.

SHARP processed Smith developed an algorithm that demosaics and super-resolutes SHARP images. The processed SHARP images have 4 times more pixel along the direction of the sensor array then the raw SHARP images. These processing results are used as a baseline to compare our results against.

3.2. Demosaicing

Training setup We train our demosaicing networks with *mean squared error* on the reconstruction of PRO images from the mosaiced simulated SHARP images. Applying the trained network to the original SHARP data then yields images from an imaginary 0.6mm not shifted sensor array. Figure 17 shows an overview of our experiment pipeline. During training for demosaicing we generally use, if not stated otherwise, a batch size of 12 and an initial learning rate of 0.001 for 30 epochs. If the validation loss does not increase for 5 consecutive epochs, the learning rate is reduced by a factor of 0.25.

Name	Description
Bottleneck ResNet	from [6]
ResNet	modified Bottleneck ResNet
N2S UNet	from [1]
UNet	our implementation

Table 1: Basic architectures in use. When refering to the UNet we usually specify the upsampling method, bilinear or transpose convolution also called upconvolution.

Our model architectures that are listed in table 1. Schematics of these are displayed in fig. 18. Our reimplementation of the Bottleneck ResNet differs as we use LeakyReLU instead of ReLU and add batch normalization layers. We use the Bottleneck ResNet as an evaluation baseline. We experiment with different ResNet architectures and settled on the one that works best. We simply call this model ResNet (table 1).

During training and fine tuning our UNet we either observed checkerboard artefacts when using transpose convolutions or over-smoothing with bilinear upsampling. These artefacts are removed by added convolutions after the upsampling and before concatenating the skip layer (fig. 23). We use 3 convolution blocks (fig. 18) after upsampling. Our UNet uses max pooling instead of convolutions for downsampling as we could not observe a difference in the output regarding this choice.

3.3. Denoising

As we neither have clean images nor pairs of noisy images, we cannot use clean target approaches or a self-supervised approach like Noise2Noise. Instead, we have to use a single self-supervised approach. We choose to adapt N2S. In the following sections we describe our basic N2S method and how we adapted it to denoising of a single channel and both SHARP channels.

Our basic adapted Noise2Self method Noise2Self divides the noisy image x into a grid of a specific grid size m. We draw an index i uniformly random from range $[0, m^2 - 1]$. The ith pixel of every cell in the grid comprises a mask. According to this mask, pixels in the noisy image are replaced by their surrounding using a convolution. This masked image $\hat{x}$ is fed into a neural net. The NN is trained on the unaltered noisy images as targets. The loss L for optimization is obtained by computing the Mean Squared Error between the noisy image and its masked version, but only where the masking applies.

$$L = MSE(x_{mask}, \hat{x}_{mask}) \tag{3}$$

Architecture for denoising The denoising architecture is the N2S Unet (fig. 18). The N2S UNet can be configured to use bilinear interpolation or transpose convolutions for upsampling. The convolutions for downsampling and upsampling are grouped with the group size set to the channel dimension. Residual connections can be enabled, such that the model is fully residual and the convolutional blocks have residuals.

Single Channel Denoising Our Single Channel Denoising method is mostly unchanged from the N2S paper. In every grid cell one pixel is masked. It is replaced by the same convolution used in N2S shown in figure 5. Our grid size for this masking is 5 by 5.

0.5	1	0.5
1	0	1
0.5	1	0.5

Figure 5: Masking Convolution

SHARP Denoising This method is for denoising of complete images from the SHARP machine. As mentioned above SHARP images have interleaved channels. We adapt the loss function and masking to account for this. For masking we define three approaches.

1 Pixel Masking One channel is uniformly randomly selected for optimization. In this channel, one pixel is masked according to the Single Channel Denoising method. Only this pixel is predicted and accounted for in the loss function. This masking assumes that the channels are noise-independent.

2 Pixel Masking One pixel in the high channel is masked as in the Single Channel method. Two pixels in the other channel overlap with this pixel. Using a uniform random drawing, one of them is masked. This means the information of the overlap between these two pixels in different layers is lost after masking. In this masking approach, the whole mask is used to compute the loss function. This masking assumes that it is enough to have a overlap of pixels to train N2S.

3 Pixel Masking Uniformly randomly select one channel to optimize. In this channel, one pixel is masked. Two pixels in the other channel overlap with this pixel. These are masked. As the original N2S masking kernel (figure 5) has a blind spot of only 1 pixel, we apply the 5x5 kernel shown in figure 12. In this approach only the mask from the channel we selected is used for masking in the loss function. This approach makes no assumptions about the noise independence between channels.

In all approaches both channels are optimized in all batches and the masking grid size used is 6 by 6.

For SHARP denoising 3 pixel masking is used, as it has the best results because of channel dependence. (See figure 21 in the appendix for a comparison.)

Bilinear upsampling We found that bilinear upsampling performes slightly better. Therefore we use it in our denoising architectures. See apendix figure 19(b/c) for a visual comparison.

0	0	0	0	0
0	0.5	1	0.5	0
1	0	0	0	1
0	0.5	1	0.5	0
0	0	0	0	0

Figure 6: Masking Convolution for big blind spot

Residual No residuals are used in our architectures for denoising, as we found that N2S training has the identity function as a local minima. Figure 7 shows a training loss trajectory of a fully residual UNet and Figure 8 the resulting residual in testing.

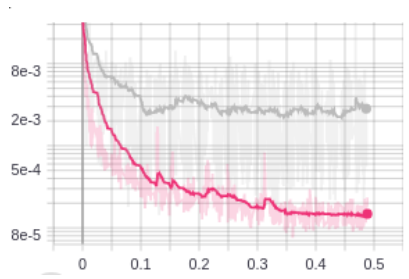
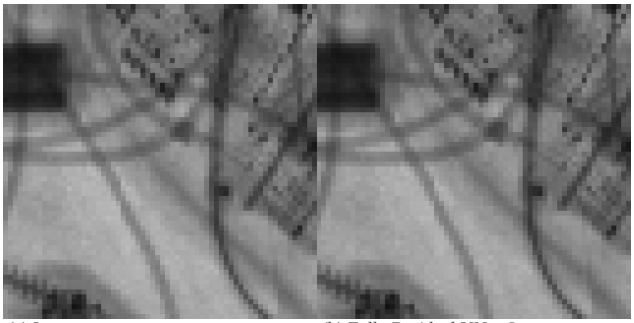


Figure 7: Local Minima Identity Function: Training Loss of residual net(grey) vs. non-residual(pink)



(a) Input (b) Fully Residual UNet Output

Figure 8: Fully residual N2S training learns identity

Noise Completion N2S learns the mean of the noise distribution. When the noise distribution is cut off, the mean in the data is shifted from the underlying mean we want to learn. This is the case for the cut-off of the distribution at zero and full intensity in our data (See left histogram in figure 10). Especially in the background, as Smith detectors are calibrated such that the peak of the background noise is set to the cut-off intensity. This leads to artefacts. For

Smith data we get much higher validation loss (figure 9) and a blurry prediction as shown in Figure 11. There are also artifacts, e.g. the white aura around the cable in the bottom center of the image.

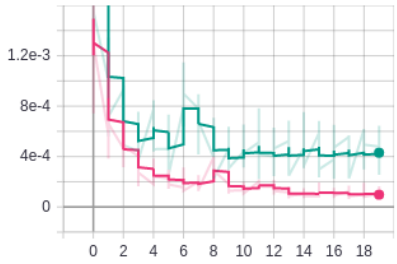


Figure 9: Validation Loss of N2S Training with noise completion(pink) and with original image statistics(green)

Therefore we implement a Noise Completion where we assign pixel in direct vicinity (5 intensity values) of a cut-off a different intensity according to a random distribution (right histogram of figure 10. This random distribution is domain specific and was chosen empirically from the histogram.

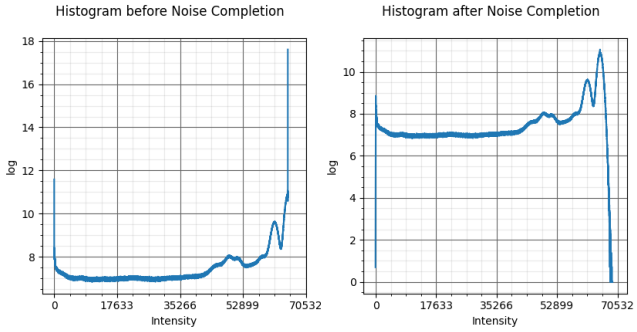
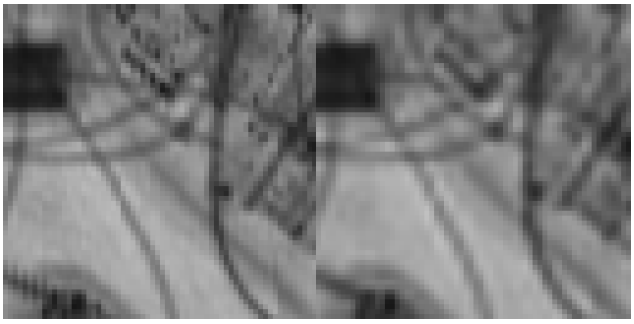


Figure 10: Histogram of all pixel intensities of the SHARP dataset, before and after Noise completion.



(a) Complemented Image Statistics (b) Original Statistics

Figure 11: Output after training with different image statistics. N2S requires complete image statistics. Note the white aura around the cable in the center.

3.4. Joint Demosaicing and Denoising

This method is for combined training of both demosaicing and denoising. We train it on PRO data by using the simulated SHARP data as input and PRO images as targets. We combine methods and architectures from Demosaicing and Denoising (fig. 17). The model architecture is the N2S UNet with bilinear upsampling and without any residuals. While residuals work well for demosaicing, they are not applicable to N2S (see 3.3).

Joint Masking The images are masked before transformation into simulated SHARP. The masking grid size is 6 by 6. We define two masking options:

Two Channel Masking: In both channels the same pixel is masked as in One Channel Denoising. Loss is computed from both channels. This is based on the assumption that the two channels are not noise-independent.

One Channel Masking: If we assume the channels are noise-independent, we can only mask one channel at a time. One channel is selected at uniform random chance and then masked as in One Channel Densoising. Loss is then only computed from this channel.

In appendix in figure 22 we compare two channel and one channel masking options for joint training. Based on these results two channel masking is used for joint training.

4. Results

4.1. Demosaicing Architectures

First we compare residual with non-residual learning. Residually learned models converge much faster during training (fig. 20). What stands out is that the N2S UNet learns much better than all other non-residual models because its highest level skip connection has the least convolutions before and after it. Therefore it is easier for this network to learn the identity.

In contrast it is difficult for the UNet(transpose/bilinear) to learn the identity what leads to much better generalization in the reconstruction of the image. The outputs of the UNets have cleaner edges and look sharper but the colorization is not that accurate. In Section we explain why colorization artifacts occur. Our UNets also denoise the images simultaneously.

The outputs of the residually trained models have very accurate colorization but are more pixelated. Also there are edge artefacts in the colorization as the model fails to fully alleviate the overlap of the wider sharp pixels.

There is a trade off between models that can learn the identity easier and therefore produce outputs robust for colorization closer to the input and models that demosaic with

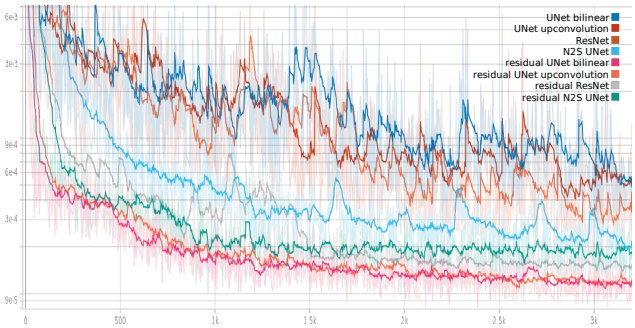


Figure 12: We can see that models that are trained on the residual learn much faster and more stable. The N2S UNet (light blue) marks an exception as it learns better than all other non-residual models while also reaching a comparable final loss value. Validation losses are comparable as seen in figure 24

Model	PSNR	SSIM	Size
Residual ResNet	**128.51**	0.9...9941	3.6MB
Bottleneck ResNet	128.08	0.9...9928	2.9MB
N2S UNet	127.47	**0.9...9952**	53MB
Transpose UNet	127.47	0.9...9889	70MB
ResNet	126.29	0.9...9748	3.6MB
Bicubic	123.70	0.9...9726	
Bilinear	123.37	0.9...9721	

Table 2: ... stands for seven 9999999 characters

higher quality but less robust regarding the colorization algorithm.

As described in section 3.3 residual learning cannot be used for denoising. Therefore the N2S UNet is the best trade off for us. As it is able to learn the demosaicing task effectively while not learning the identity during denoising. Of all non residually trained models the N2S UNet has the best performance (fig. 20).

Colorization Problem Optimizing the prediction on the two channel image does not imply a good RGB image output (fig. 13). Very small not directly visible errors can have a large impact on the colorization. The RGB color is influenced by the ratio of the high and low channel. Therefore, very small changes in the output can already throw these ratios off. This problem is not that noticeable for residual models (see fig. 20), as its output is by nature closer to the input and therefore in the correct value range.

4.2. Demosaicing Evaluation

We compare all models listed in table 1, demosaicing with bicubic interpolation and an algorithm developed by Smith. We use peak signal to noise ration (PSNR) and structural similarity index metric (SSIM) between the PRO image and the reconstructed PRO image from the simulated SHARP (table 2).

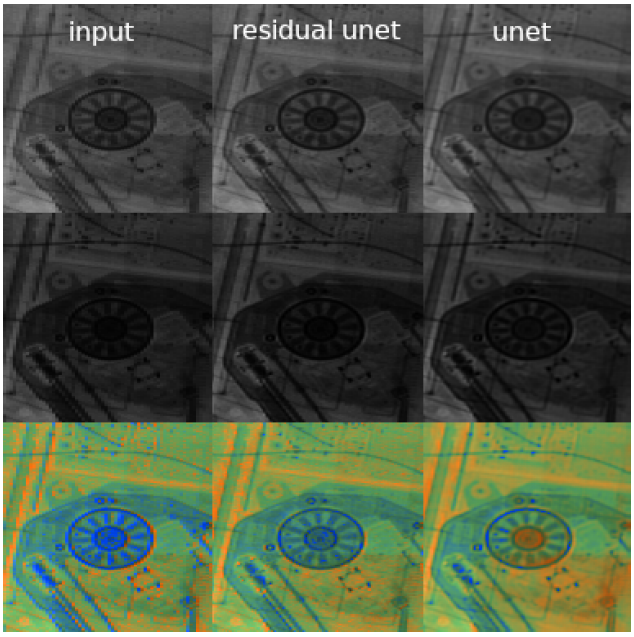


Figure 13: Top: high energy channel, middle: low energy channel, bottom: RGB conversion. Even though the outputs of both UNets look like a very good demosaicing result the RGB predictions differ significantly. There are many artefacts in the RGB conversion of the UNet output. The reason for this is the way the colors are calculated from the two-channel image.

Figure 14 compares colorized outputs of all evaluated methods. Our residual ResNet has the output that is closest to the PRO image of the same patch. While this is expected regarding out training methods this might not be the best achievable output overall. Smiths output looks more appealing overall as the colors are smooth and objects on in the image are easy to recognize. While out methods try to reconstruct a potentially lost image from the mosaiced their algorithm also performs enhancements designed with the use case in mind. Bicubic upsampling produces a surprisingly good result except that it has edge artefacts. Because we upsample each channel individually and the SHARPs sensor array is interleaved bicubic interpolation cannot entangle them.

While both ResNets make the top of the PSNR rating their visiual output is not as appealing. Looking closely at the wires going through the image we can see that their edges are pixelated in comparison to the UNets output. UNets main advantage is the ability to generalize SHAPE and correctly reconstruct them. Noise is mostly removed and edges are clean and nicely interpolated without pixelation. SSIM is not very assertive in our case, as the two channel outputs of our models are near perfect reconstructions of the

PRO images. The differences observed in the colorized images is not reflected in the SSIM. However, the rankind of models with SSIM correlates with the model performance.

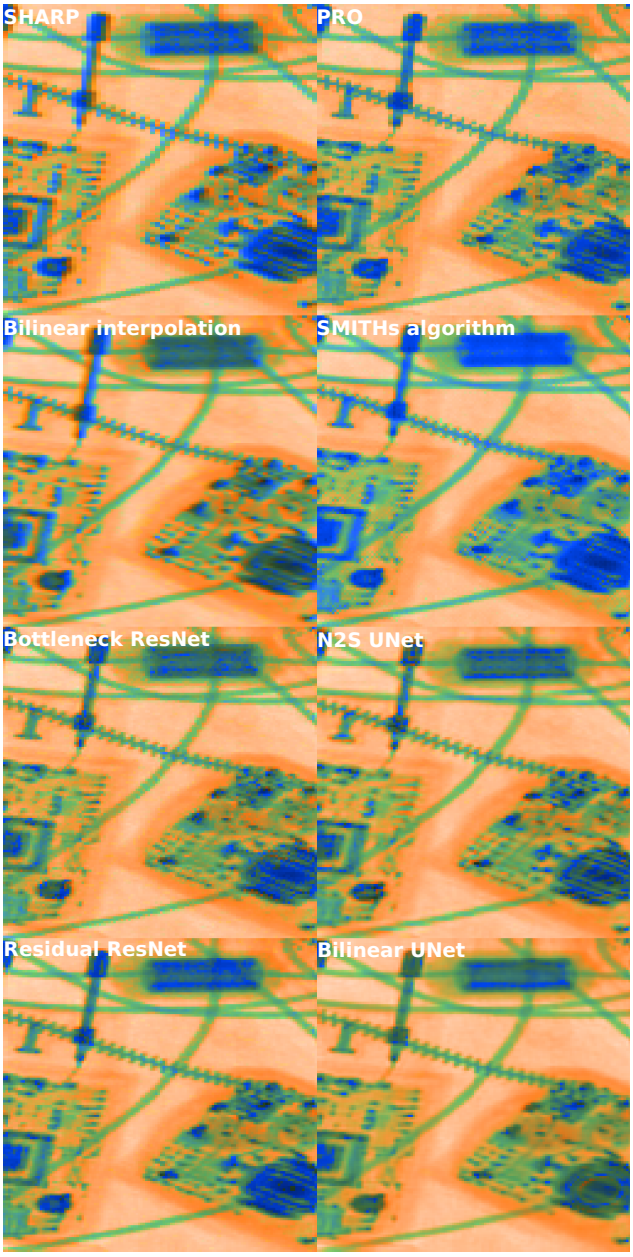


Figure 14: Evaluation of demosaicing algoithms on SHARP data. The image of the PRO detector of the same patch was attached for reference. Smith algorithm has a higher resolution because it also super-resolutes the image. We notice checkerboard pattern artifacts in the output of Smiths algorithm. As expected the residual ResNet has the output that is closest to the PRO regarding colorization while being sharper at the same time.

4.3. Denoising

In figure 15 we show results from SHARP denoising, one channel denoising and compare these to BM3D.

One Channel Denoising One channel denoising works well, but we find that BM3D is able to keep more fine pixel-scale structures. That is expected, as N2S assumes the signal to be pixel-wise dependent and predictable. For very small structures this is not the case. Meanwhile BM3D has the information of the current pixel available. Note that BM3D also needs the variance of the noise as input, while N2S does not have this information.

SHARP Denoising SHARP denoising does not work as well as one channel denoising. It struggles with apparently not completely independent pixel in 1 and 2 pixel masking,

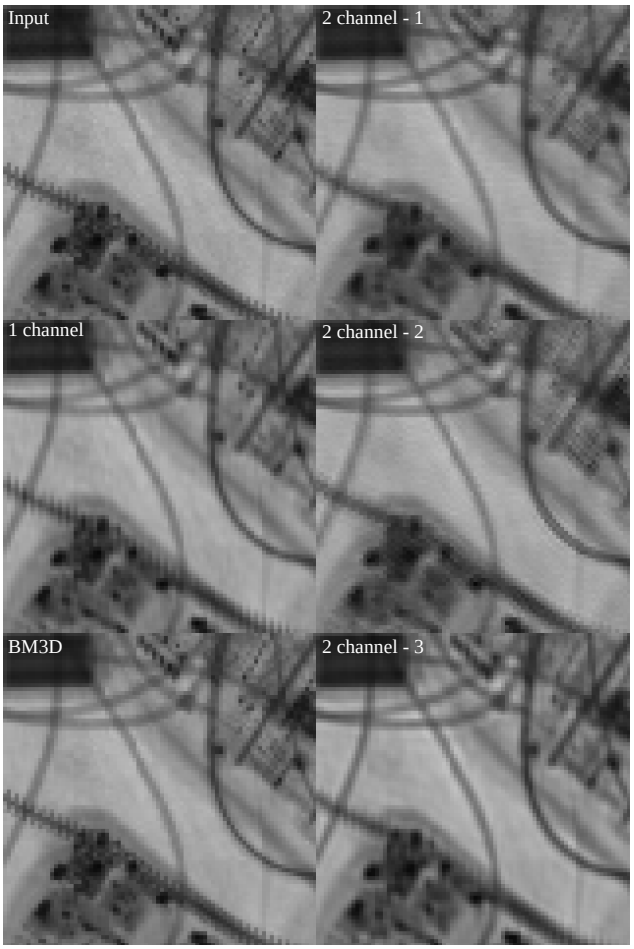


Figure 15: Denoising Results: Input, Single Channel Denoising, BM3D and SHARP denoising with 1,2 and 3 pixel masking. One channel denoising performes better than SHARP denoising. BM3D retains structures that are not possible for N2S to predict.

which lead to small artifacts. Considering the completely masked 3 pixel masking we find it to have similar results to one channel denoising, but more blurry output. This can be explained by the greater blind spot of 3 pixel masking.

4.4. Joint Demosaicing and Denoising

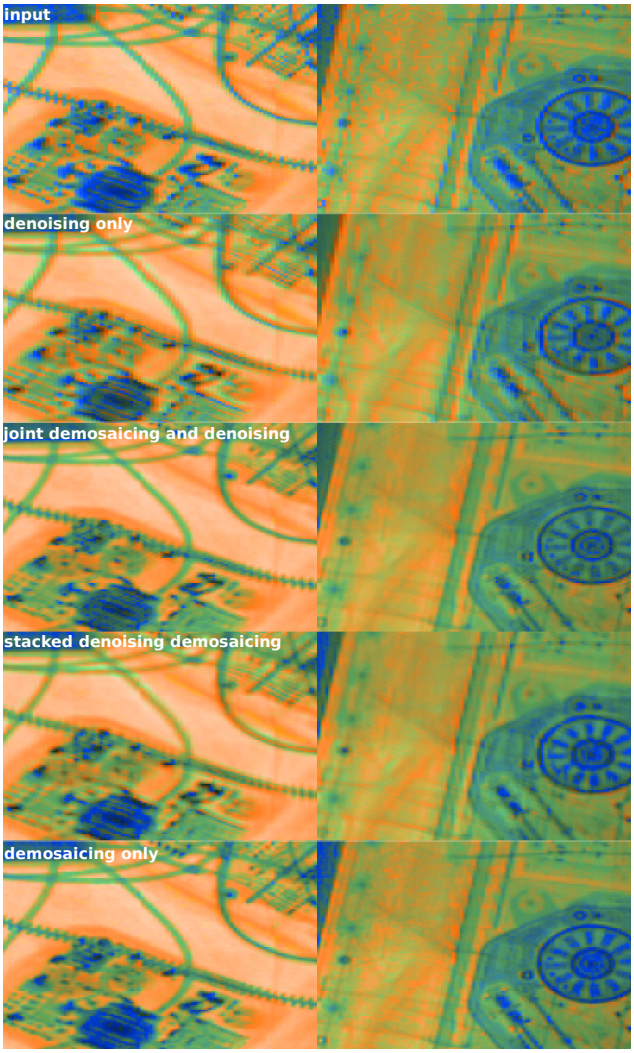


Figure 16: Joint training evaluated on SHARP data. From top to bottom: input, only denoising, joint, stacked, demosaicing. While denoising works well considering the high and low images, we see that the RGB images still have the channel shift, which is to be expected, and therefore give colourful edges. The joint model performes better than the stacked model.

We compare the joint model visually to denoising and demosaicing respectively standalone and stacked (Figure 16). We find that the joint model performes better than stacking models. This is due to drawbacks in interleaved channel masking, which the joint model does not have as the masking is done on perfectly overlapping pixel. Compared to just denoising joint has the colors on edges correct after color conversion, because it can solve the shift in channels. Just demosaicing on the other hand has the noise still visible compared to joint training.

4.5. Inference Runtimes

We compare inference runtimes between our models and to the denoising reference BM3D. The joint model has the same runtime as denoising because they use the same architecture. BM3D is slower than our denoising.

Method	Inference Runtime [s]
BM3D (single channel)	0.49
N2S single channel denoising	0.21
N2S two channel denoising	0.21
Res-Resnet demosaicing	0.49
N2S Joint	0.21
Stack (N2S, Res-Resnet)	0.70

Table 3: Inference runtimes of our methods and BM3D reference.

5. Conclusion

Not all demosaicing methods work equally well on X-Ray images. While all give good visual results on the single channel views, they perform differently when comparing the RGB conversions. Networks with fully residual architecture have the best performance regarding colorization. We present a ResNet architecture that performs slightly better than the baseline Bottleneck ResNet. Not fully residual models such as out UNet give the best structural reconstruction of the image while they are not accurate enough for correct colorization. We think the best result regarding colorization could be achieved by training non residual UNet that directly predicts a RGB image. We did not test out but assume that directly predicting a demosaiced 3 color RGB image from the two channel input has to potential yield better results and also does not require residual learning.

We introduced methods to apply Noise2Self to X-Ray images with interleaved channels and evaluated them on real world data. For interleaved two channel denoising we defined three masking options, based on different assumptions. We found learning the residual is a significant local minima for N2S. Furthermore, we showed that meeting assumptions on image statistics, namely that noise should be pixel-wise independent and zero-mean and the signal pixel-wise dependent are important to achieve good perfomance in N2S. As the image statistics do not meet these requirements, we introduced Noise Completion.

Joining tasks improves performance in terms of accuracy as well as inference runtime.

References

[1] J. Batson and L. Royer. Noise2self: Blind denoising by self-supervision. *arXiv preprint arXiv:1901.11365*, 2019.

[2] A. Krull, T.-O. Buchholz, and F. Jug. Noise2void-learning denoising from single noisy images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2129–2137, 2019.

[3] J. Lehtinen, J. Munkberg, J. Hasselgren, S. Laine, T. Karras, M. Aittala, and T. Aila. Noise2noise: Learning image restoration without clean data. *arXiv preprint arXiv:1803.04189*, 2018.

[4] N.-S. Syu, Y.-S. Chen, and Y.-Y. Chuang. Learning deep convolutional networks for demosaicing. *arXiv preprint arXiv:1802.03769*, 2018.

[5] R. Tan, K. Zhang, W. Zuo, and L. Zhang. Color image demosaicking via deep residual learning. In *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, pages 793–798, 2017.

[6] D. Verma, M. Kumar, and S. Eregala. Deep demosaicing using resnet-bottleneck architecture. In *International Conference on Computer Vision and Image Processing*, pages 170–179. Springer, 2019.

[7] M. Weigert, U. Schmidt, T. Boothe, A. Müller, A. Dibrov, A. Jain, B. Wilhelm, D. Schmidt, C. Broaddus, S. Culley, et al. Content-aware image restoration: pushing the limits of fluorescence microscopy. *Nature methods*, 15(12):1090–1097, 2018.

[8] R. Zhou, R. Achanta, and S. Süsstrunk. Deep residual network for joint demosaicing and super-resolution. *ArXiv*, abs/1802.06573, 2018.

A. Appendix

We append pipeline, model architectures in detail and further visual model evaluations.

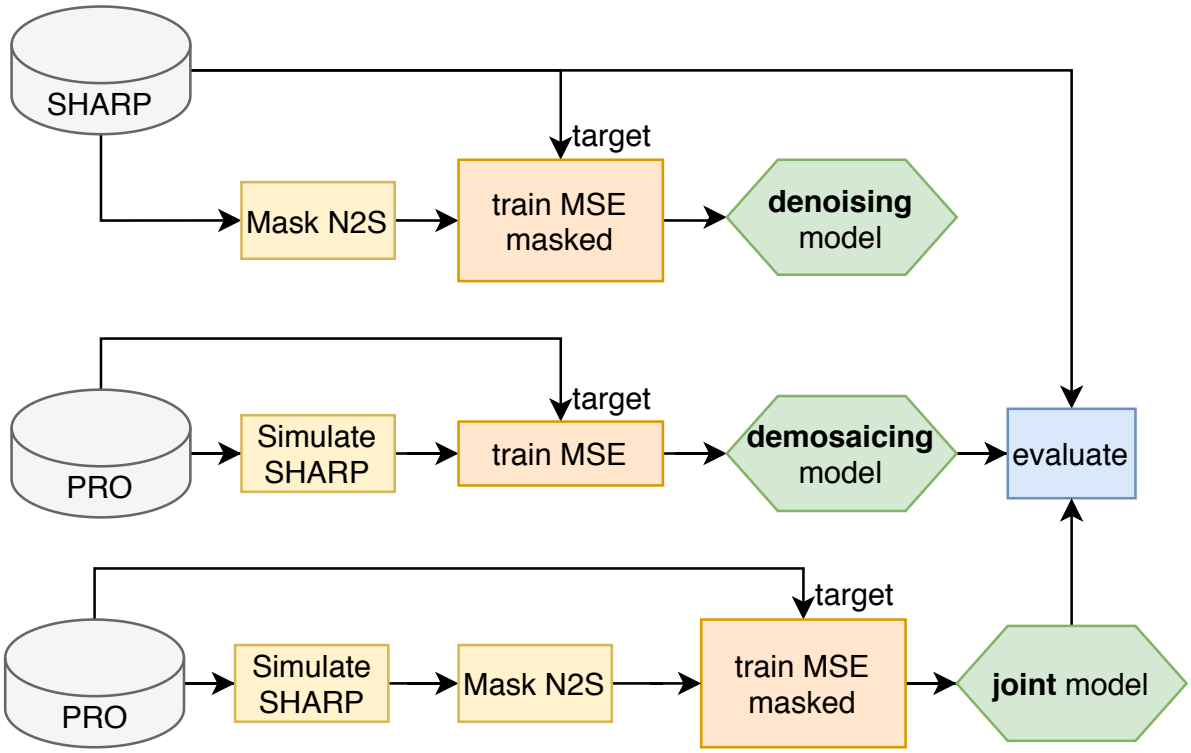


Figure 17: Schematic describing how we use our datasets for training and evaluation.

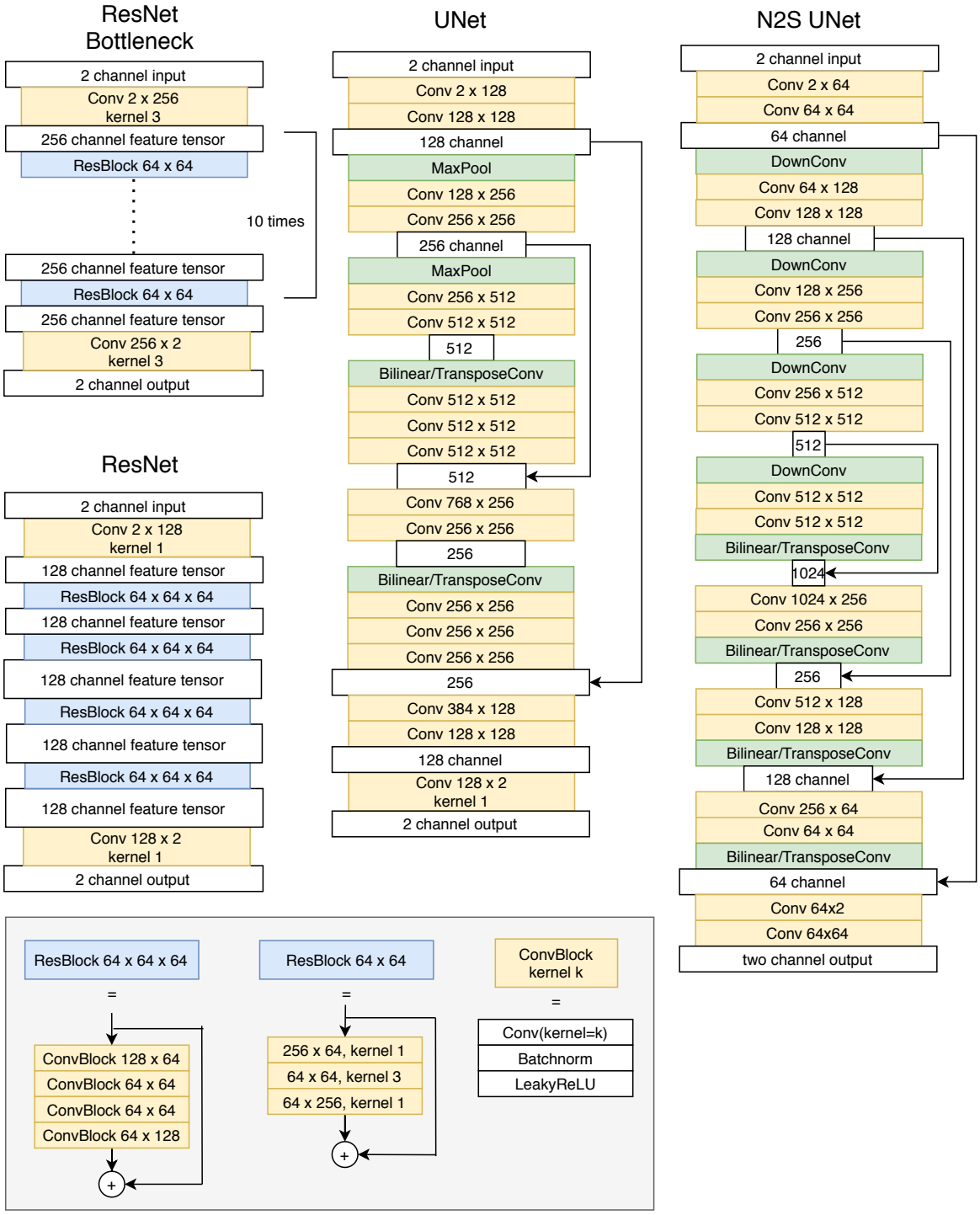
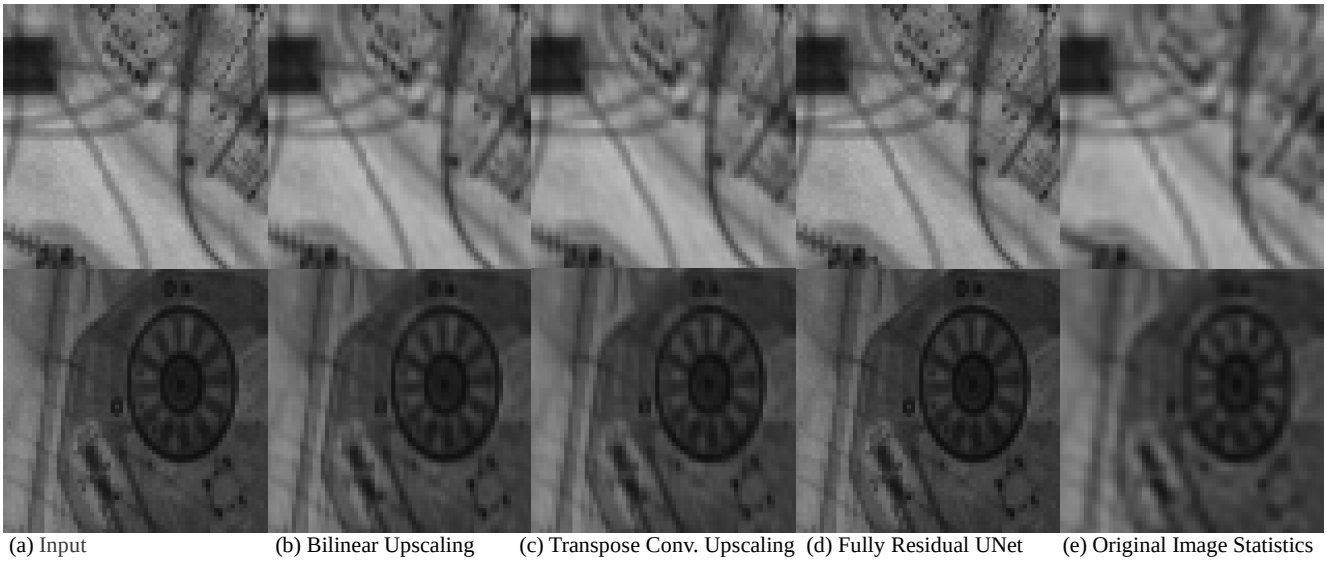


Figure 18: Different kinds of architectures that we trained on both tasks, denoising and demosaicing. White boxes are intermediate feature tensors. Yellow boxes are convolutions, blue boxes are multiple convolutions that have a skip connection over them. We call these residual blocks. Green boxes are up or downsampling layers. A skip connection pointing into a white box means concatenation. The number in the white box is the number of channels before concatenating. Not the three convolutional blocks between each upsampling and concatenation in the UNet. We explain these in fig. 23

.



(a) Input (b) Bilinear Upscaling (c) Transpose Conv. Upscaling (d) Fully Residual UNet (e) Original Image Statistics

Figure 19: One Channel Denoising applied to the high channel of SHARP. (b) and (c) show the differences in bilinear versus transpose convolution as upscaling stages. (d) A fully residual UNet learns the identity function as a local minimum. (e) Without noise complementation N2S gives us a blurry output.

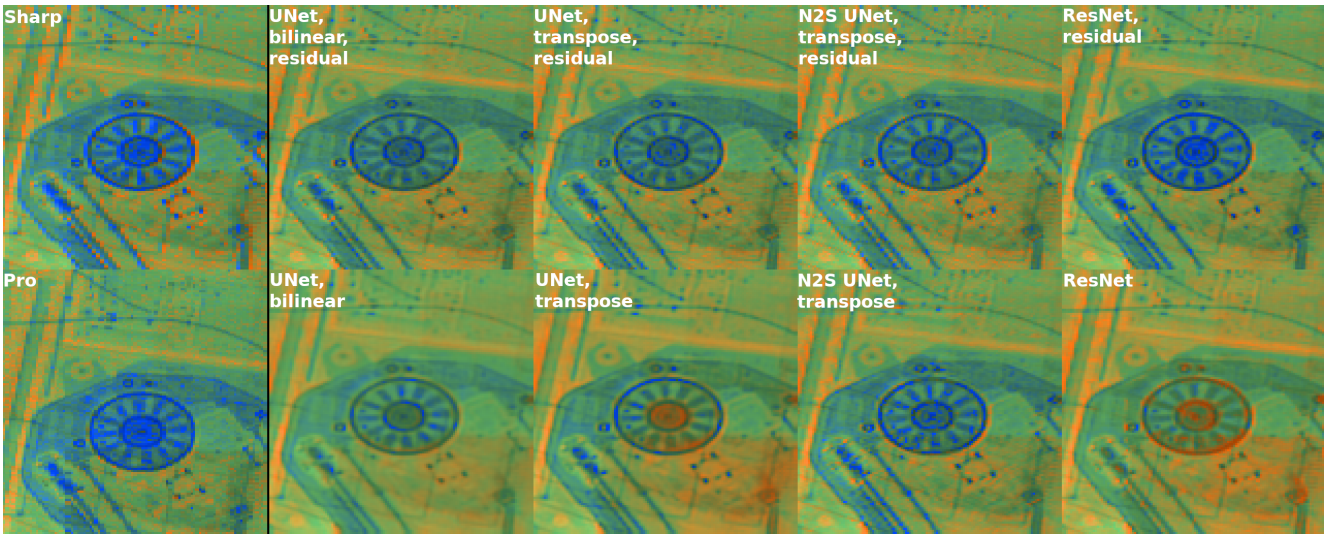


Figure 20: We compare fully residual models (top) against not fully residual models (bottom). The first column, separated by a black line shows the raw SHARP and PRO patches of the respective areas. It is directly visible, that the colors for fully residual are much closer to the ones in the PRO image. One can also observe, that residual models produce pixelated outputs. Due to its additive nature it is more difficult to smooth out the edges in the input. Non residual models generalize and reconstruct the shapes well such that edges are smooth and clean. This is an inherent effect of the models compressing nature.

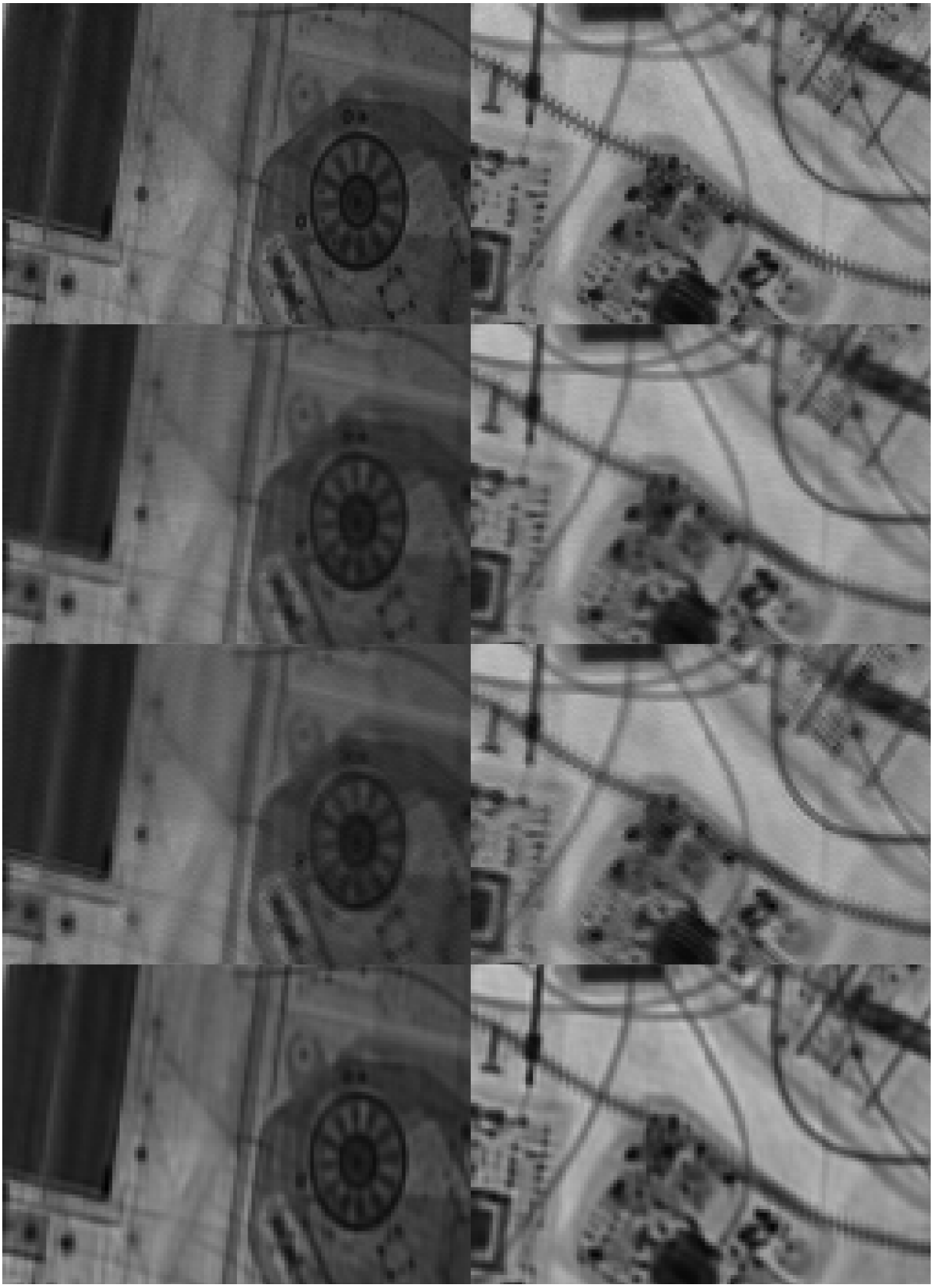


Figure 21: Two Channel Denoising of SHARP data (high channel). From top to bottom: Input, 1,2 and 3 pixel masking.

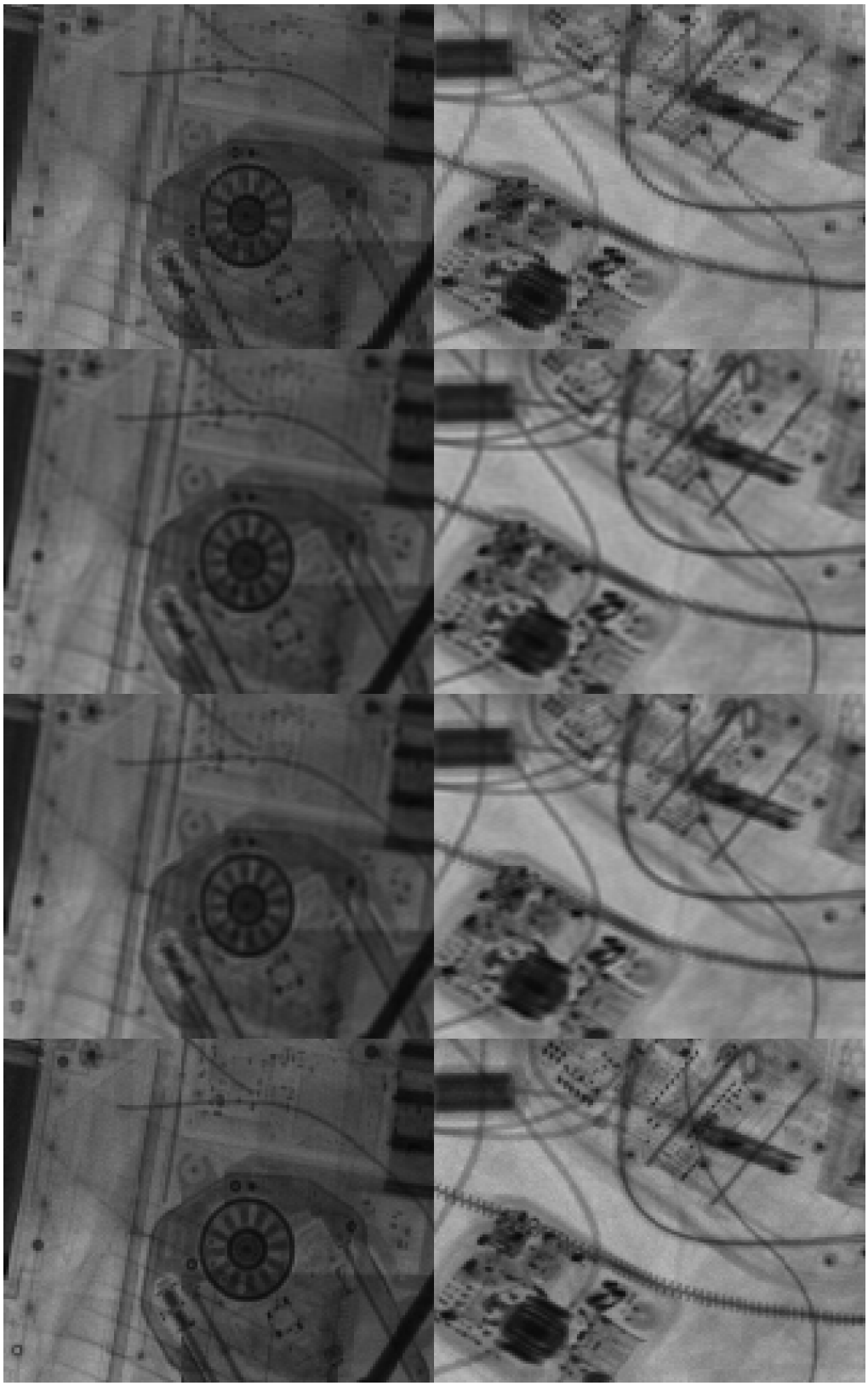


Figure 22: High channel from joint training. From top to bottom: Artificial SHARP input, one channel masking output, two channel masking output and ground truth. One channel masking is more blurry, while two channel masking shows more detail (for example at the zipper or at electronics parts).

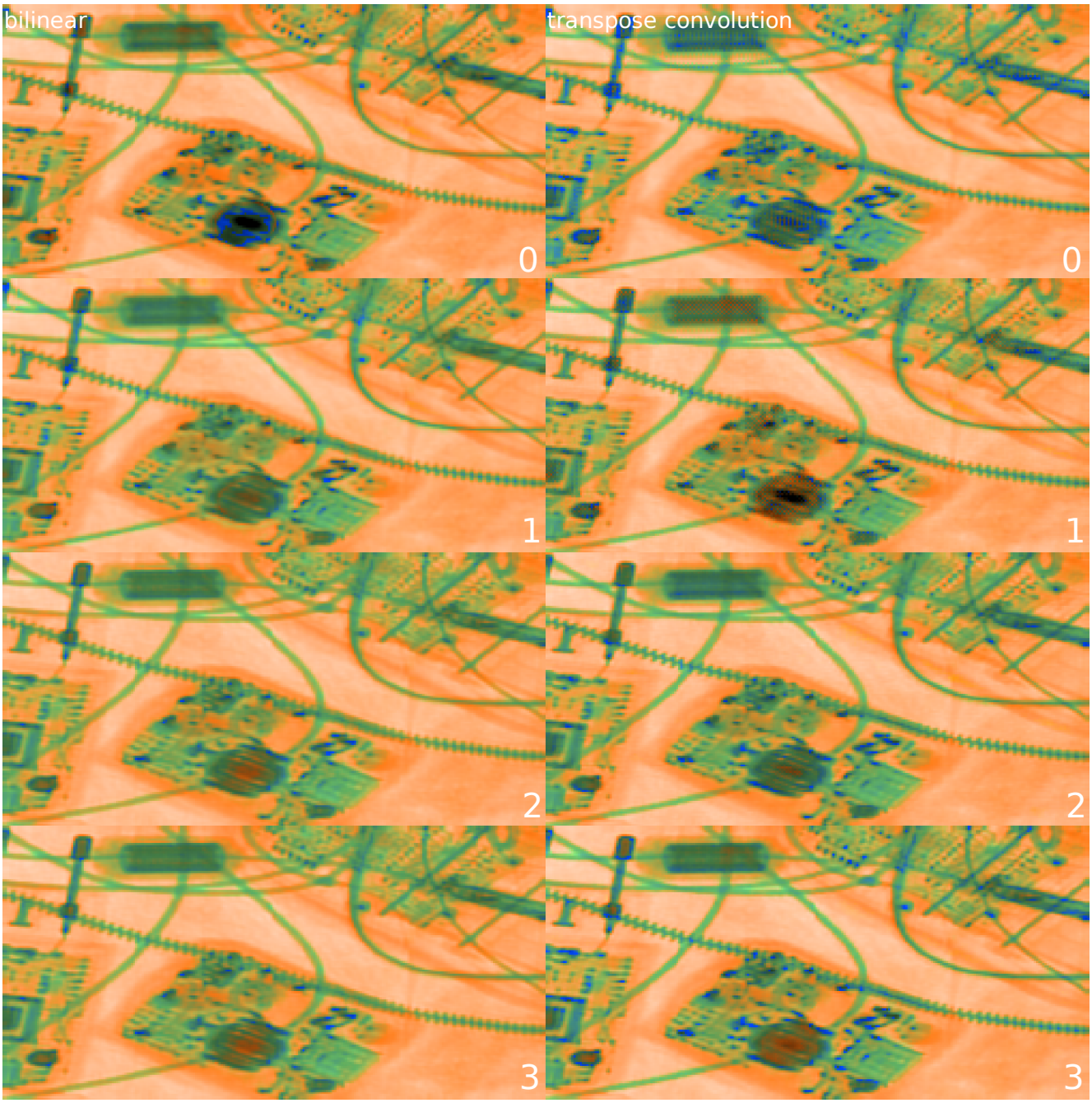


Figure 23: We use our UNet with bilinear interpolations and transpose convolutions and insert 0, 1, 2 and 3 additional convolutions between upsampling and concatenating to the skip connection. The bilinear UNets output slightly sharpens by adding one more convolution. But then does not improve anymore. The checkerboard artefacts from the transpose convolution operation are gradually removed by introducing more convolutions. That is why we parameterize all our UNets with 3 additional convolutions as visualized in fig. 18.

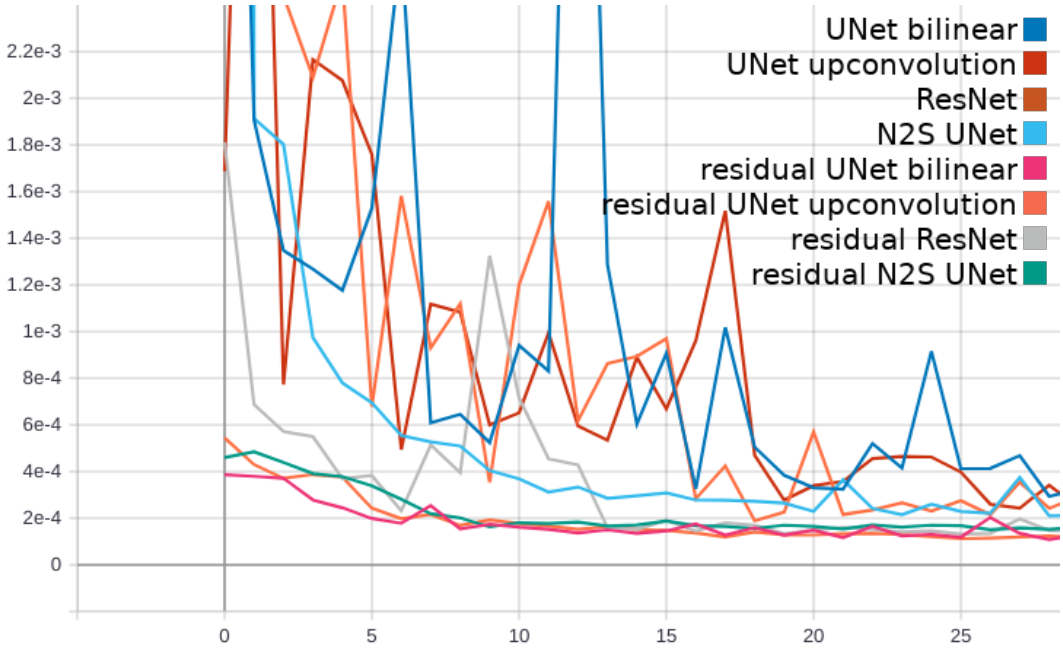


Figure 24: Validation loss of different models during training.