Class-Aware Fully-Convolutional Gaussian and Poisson Denoising

Tal Remez, Member, IEEE, Or Litany, Member, IEEE, Raja Giryes, Member, IEEE, and Alex M. Bronstein, Member, IEEE

Abstract—We propose a fully-convolutional neural-network architecture for image denoising which is simple yet powerful. Its structure allows to exploit the gradual nature of the denoising process, in which shallow layers handle local noise statistics, while deeper layers recover edges and enhance textures. Our method advances the state-of-the-art when trained for different noise levels and distributions (both Gaussian and Poisson). In addition, we show that making the denoiser class-aware by exploiting semantic class information boosts performance, enhances textures and reduces artifacts.

Index Terms—Image denoising, Gaussian noise, Poisson noise, Video denoising, Deep learning, Fully-convolutional networks, Class-aware denoising.

I. INTRODUCTION

Denoising of additive white Gaussian noise and shot noise (Poisson-distributed) are fundamental problems in image enhancement. The Gaussian model is typically used for describing thermal noise and approximating shot noise in high- and medium-light imaging. In the low-light regime, which is largely dominated by shot noise, the Gaussian model fails to accurately represent this noise, and the more accurate Poisson model is used instead. In addition to image denoising purposes, it has been shown in [2], [3], [4], [5], [6] that a good Gaussian denoising algorithm may serve as a prior for efficiently solving many other image processing inverse problems such as deblurring or inpainting within a single generic framework.

In recent years, state-of-the-art in denoising of images has been achieved by techniques based on artificial neural networks [7], [8], [9], [10], [1], [11]. Following this promising trend, we propose a new fully-convolutional network architecture for image denoising that advances the state-of-the-art for both types of noise and most noise levels. In contrast to previous methods, our network estimates the noise gradually through the composition of noise estimators extracted at intermediate layers.

The latter fact makes intermediate results directly useful and allows a well timed termination of the denoising process. In addition, it allows to inspect the inner workings of the network and gain an insight into its operation. Interestingly, the network produces a monotonically decreasing error without being explicitly trained to do so. Furthermore, being fully-convolutional it does not suffer from blocking artifacts characterizing non-overlapping patch-based approaches, or from the additional complexity required when overlapping patches are processed and averaged to improve quality and avoid the blockiness [12], [13], [14].

Having a good denoiser at hand, one may ask how it can be improved even further. Patch-based image denoising theory suggests that existing methods have practically converged to the theoretical bound of the achievable performance [15], [16], [17], [18]. Yet, it turns out that two possibilities to break this barrier still exist. The first is to use larger patches which despite the law of diminishing return for complex patches described in [15], has been proven useful in [7], where the use of 39 × 39 patches allowed to outperform the popular BM3D denoising algorithm [19]. The second possibility is to use a better image prior, such as narrowing down the space of images to a more specific class. These two possibilities are not mutually exclusive, and indeed we show an approach that exploits them both.

To exploit larger patches, our network has a receptive field of size 41 × 41, which is bigger than the common existing practice. The convolutional architecture is instrumental to prevent the network from becoming prohibitively large. To exploit a narrower image subspace, we draw inspiration from previous studies showing the benefit of designing a strategy for a specific class of images [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30], [31]. To this end, we propose a class-aware denoising framework, in which the denoiser is fine-tuned to best fit a particular class of images. At inference, the class information can be provided manually by the user, for example, choosing face denoising for cleaning a personal photo collection, or automatically via a classification algorithm. We show that class-awareness indeed boosts performance significantly when the class is given by an oracle. Moreover, combining our denoiser with an off-the-shelf classifier, fine-tuned to our classes, instead of the oracle leads to comparable performance.

Contribution. Our contribution is threefold:

1) We propose a novel fully-convolutional neural-network architecture that is comparable to the state-of-the-art for Gaussian and Poisson image denoising and Poisson video denoising.

2) Our network design grants easy access to the noise being removed at intermediate layers, allowing interesting insights into its inner workings.

3) We demonstrate an additional boost in performance when classifying the input image before routing it to a class-specific denoiser.

While this paper focuses on denoising, our methodology can be easily extended to much broader class-aware image enhancement and restoration problems, rendering it applicable to
many image processing and computer vision tasks. Preliminary results of this paper for Gaussian image denoising have been presented in [32]. We show how the proposed deep architecture can be further used for Poisson denoising, demonstrating examples of simulated and real low-light images and videos.

The rest of the paper is organized as follows: Section II surveys related work. In Section III, we describe our denoising architecture and show how it can be made class-aware. Section IV includes a description of our implementation. Section V presents an experimental evaluation of class-agnostic and class-aware denoising for Gaussian and Poisson noise.

II. RELATED WORK

Numerous methods have been proposed for removing Gaussian noise from images, including k-SVD [33], non-local means [34], BM3D [19] non-local k-SVD [35], field of experts (FoE) [36], Gaussian mixture models (GMM) [37], non-local Bayes [38], global image denoising [39], nonlocally centralized sparse representation (NCSR) [40] and simultaneous sparse coding combined with Gaussian scale mixture (SSC-GSM) [41].

A popular strategy for recovering images contaminated by Poisson noise relies on variance-stabilizing transforms (VST), such as Anscombe and Fisz [42], [43], [44], which convert Poisson noise to be approximately Gaussian with unit variance. Thus, it is possible to perform Poisson denoising using Gaussian denoisers, e.g., by VST+BM3D [45]. Recently, it has been shown that it is possible to improve this method by iteratively applying the VST and the Gaussian denoiser (I+VST+BM3D) [46]. However, such approaches are limited as their performance deteriorate significantly for low intensities. Another approach is to cast the Poisson denoising problem to a Gaussian one using the plug-and-play framework [5] as suggested in the P^3IP method [47].

Alternatively, there are methods that are applied directly to the Poisson noisy data such as the non-local sparse PCA (NLSPCA) technique [48] or the sparse coding based Poisson denoising algorithm (SPDA) [49].

A popular strategy improving the performance of many Poisson denoising algorithms is binning [48]. Instead of processing the noisy image directly, a low-resolution version of the image with higher SNR is generated by aggregation of nearby pixels. A Poisson denoising technique is then applied followed by simple linear interpolation to recover the original high resolution image. The binning technique trades-off spatial resolution and SNR, and has been shown to be useful in the very low SNR regimes.

Most aforementioned techniques are designed based on some properties of natural images such as the recurrence of patches at different locations and scales, or their sparse representation in some (possibly trained) dictionary. In the past few years, however, the state-of-the-art in image denoising has been achieved by techniques based on artificial neural networks. The first such method is the multilayer perceptron (MLP) proposed in [7]. It is based on a fully connected architecture and therefore requires a long training time, a large amount of memory, and has a high arithmetic complexity at inference. In [8] (TRDPD) and [9] (TNRD), nonlinear diffusion based neural networks have been proposed for Poisson and Gaussian denoising respectively. Another work [10], proposes a neural network based on a deep Gaussian Conditional Random Field (DGCRF) model. More recently, in parallel to our work, a convolutional neural network with batch normalization and without intermediate noise estimations has been proposed (IRCNN) [1].

Designing a strategy for a specific class of images has been shown to be beneficial for many image enhancement applications. For example, in [20], the authors set a bound on super-resolution performance and showed it can be broken when a
face-prior is used. In [21], a compression algorithm for facial images was proposed. Face hallucination, super-resolution, and sketch-photo synthesis methods were developed by [22]. In [23], the authors showed that given a collection of photos of the same person, it is possible to obtain a more realistic reconstruction of the face from a blurry image. In [24], [25], class labeling at a pixel-level was used for the colorization of grayscale images.

The closest class-specific approach to our work is the one proposed in [26], [27], [28], [29]. It shows improvement in image restoration tasks (denoising and deblurring) by adapting the reconstruction algorithm to a specific class of images. The proposed strategy is patch based and assumes a priori knowledge of the class. Our solution circumvents these two limitations by (i) using a fully convolutional neural network that processes the image as a whole; and (ii) a classification network that can detect the class of the image.

III. METHOD

A. Class-agnostic network architecture

Our network structure was designed according to the following principles: First, it is fully convolutional to facilitate denoising images of varying size. This eliminates the use of image patches which, in turn, entails carefully choosing an aggregation procedure for the denoised overlapping patches. Two advantages achieved by such a paradigm are the relatively small number of parameters and fast execution time as demonstrated in the experiments in Section V. A second requirement was to have a gradual denoising process since this has been shown by [50], [51], [52] to yield better denoising.

Our network architecture is presented in Fig. 2. In each layer, we perform 63 convolutions (marked in light blue) followed by a ReLU and forwarded to the next layer. We additionally perform a single $3 \times 3$ convolution, whose weights are learned together with the other parameters of the network. This convolution yields a matrix of the same size as the input image (marked in dark blue in the figure). This matrix is added to the noisy input and is not forwarded to the next layer. We refer to these single channel matrices as noise estimates because their sum cancels out the noise. In all layers we use convolutions with a fixed size of $3 \times 3$ and stride 1. In Section V-B, we visualize the noise estimates at different layer depths to gain insights into what the network has learned.

As the sum of the layers is added to the noisy input image in order to produce the clean image, the effect these layers produce is noise removal instead of directly estimating the clean signal. The fact that only a fraction of the noise is removed in each layer of the network makes the propagation of the information easier as the signal is recovered gradually and not all at once. This provides some kind of boosting in the recovery, which is shown to be helpful also in the context of conventional image denoising [50], [51].

In the case of Poisson noise, we do not apply the binning or the Anscombe transform as the common practice suggests [35], [48], [53], since this pre-processing did not show any performance improvement of the network.

Fig. 2. Network architecture. Each layer performs 63 convolutions followed by a ReLU and forwarded to the next layer (light blue). An additional convolution yields a matrix (dark blue), which is added to the input image. Tensor sizes are listed as Width $\times$ Height $\times$ #Channels.

B. Class-aware denoising

Class-aware denoising requires handling each class in a specialized manner. Our method comprises two stages: First, the image is classified into one of the supported classes using a classifier. Then, it is fed into a class-specific denoising network of the chosen class. In all class-aware experiments, we used the following five semantic classes: face, pet, flower, living-room, and street.

As a classifier, we used the pretrained weights of the convolution layers of VGG16 [54] and trained four additional fully connected layers of sizes 1024, 1024, 1024, and 5 with ReLU and drop-out, and a soft-max layer at the very end.

Each of our class-specific denoisers has the same architecture as described before and is trained using ImageNet images [55] from a specific (semantic) class. In the experiments we show that the described two-stage system boosts performance significantly when the class is given by an oracle, and that replacing the oracle with a classifier does not deteriorate the gain in the performance.

IV. IMPLEMENTATION DETAILS

In all experiments we used denoising networks with 20 layers implemented in TensorFlow [56] and trained for 160K mini-batches on a Titan-X GPU. The mini-batches contained 64 patches of size $128 \times 128$. Color images were converted to YCbCr, and the Y channel was used as the input grayscale image after it had been scaled and shifted to the range $[-0.5, 0.5]$ and contaminated with simulated noise. During training, image patches were randomly cropped and flipped about the vertical axis. As the receptive field of the network is of size 41, the outer 21 pixels of the input suffer from convolution artifacts. To avoid these artifacts in training, we calculate the loss of the network only at the central part of each patch used for training, not taking into account the outer 21 pixels in the calculation of the loss. At test time, we pad the image symmetrically by 21 pixels before applying our network and remove these added shoulders from the denoised image. We used an $\ell_2$-loss in all experiments. Training was performed...
class-agnostic denoising. For MLP, TNRD and IRCNN we used
methods: (i) BM3D [19]; (ii) multilayer perceptrons (MLP)

set as well as on a real low-light image. In all experiments
method on Gaussian and Poisson noise using the following test
A. Class-agnostic denoising

1) Gaussian denoising: We compare our network perfor-
mance at removing Gaussian noise with standard deviation
values between 10 and 75 against the following leading
methods: (i) BM3D [19]; (ii) multilayer perceptrons (MLP)
[7]; (iii) TNRD [9]; and (iv) IRCNN [1] using the pretrained
models provided online. For MLP, TNRD and IRCNN we used

the publicly available pre-trained, noise-level-specific models.
We tested our denoising algorithms on a test set of 1K images
from PASCAL [58]. Table I summarizes the performance in
terms of average PSNR.

Note that the IRCNN pre-trained network for Gaussian
noise provided by the authors of [1] is trained on the BSD
dataset and not PASCAL. However, as shown in [61], our
network still gets better performance than IRCNN even when
the latter is trained on PASCAL. In the following sections
we show that this result is consistent when we re-train the
IRCNN network for other scenarios (such as Poisson noise)
and compare to our network.

Figure 3 emphasizes the statistical significance of the
improvement achieved by our method for Gaussian denoising
with \( \sigma = 25 \). It compares the gain in performance over
BM3D achieved by our method with the one achieved by MLP,
TNRD and IRCNN. The plot clearly visualizes the consistent
improvement in PSNR achieved by our method. Additionally,
for \( \sigma = 25 \), our method outperforms competing methods on
82% of the images, whereas IRCNN, MLP, BM3D, and TNRD
win on 13%, 4%, 1% and 0% respectively.

Next, we tested how well our network performs on the
widely used test set of 68 images selected by [59] from
Berkeley segmentation dataset [60]. Table II shows that our
method generalizes well to this dataset, for all \( \sigma \) values.

2) Poisson denoising: We evaluated our method on peak
values ranging from 1 up to 30 and compared its performance
to leading methods. Whenever code for other techniques was
not publicly available, we evaluated our method on the same
test set they had been tested on, and compared our performance
with their reported scores. We reimplemented IRCNN and
trained it on the exact same dataset that we used with our
network. We tried it with and without batch normalization. We
found that batch normalization reduced its PSNR performance
by 0.15 dB on average for the different peak values. A similar
phenomenon was observed in [61] in the context of image
burst denoising. Therefore we only present the results for the
architecture that does not have batch normalization.

Similarly to the Gaussian noise, we first evaluated our
method on 1K test images from PASCAL [58]. Table III
summarizes performance in terms of average PSNR. Figure
4 shows a profile of the gain in performance of our method
with respect to I+VST+BM3D [46] for all peak values.

Next, we tested our method on the 68 test images from
the Berkeley dataset [60]. Note that we did not fine-tune our
network to fit this dataset but rather used it after it had been
trained on PASCAL. Results are summarized in Table IV.

Finally, we evaluated our method on the standard 10 image

TABLE I
GAUSSIAN DENOISING ON PASCAL. AVERAGE PSNR VALUES ON A 1K IMAGE TEST-SET.

<table>
<thead>
<tr>
<th>( \sigma )</th>
<th>10</th>
<th>15</th>
<th>25</th>
<th>35</th>
<th>50</th>
<th>65</th>
<th>75</th>
</tr>
</thead>
<tbody>
<tr>
<td>BM3D</td>
<td>34.26</td>
<td>32.10</td>
<td>29.62</td>
<td>28.14</td>
<td>26.61</td>
<td>25.64</td>
<td>25.12</td>
</tr>
<tr>
<td>MLP</td>
<td>34.29</td>
<td>–</td>
<td>29.95</td>
<td>28.49</td>
<td>26.98</td>
<td>26.07</td>
<td>25.54</td>
</tr>
<tr>
<td>TNRD</td>
<td>–</td>
<td>32.35</td>
<td>29.90</td>
<td>–</td>
<td>26.91</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>IRCNN</td>
<td>–</td>
<td>32.66</td>
<td>30.25</td>
<td>–</td>
<td>27.22</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Ours</td>
<td><strong>34.87</strong></td>
<td><strong>32.79</strong></td>
<td><strong>30.33</strong></td>
<td><strong>28.88</strong></td>
<td><strong>27.32</strong></td>
<td><strong>26.30</strong></td>
<td><strong>25.75</strong></td>
</tr>
</tbody>
</table>

TABLE II
GAUSSIAN DENOISING ON 68 IMAGE SET FROM [59]. AVERAGE PSNR VALUES ARE PRESENTED.

<table>
<thead>
<tr>
<th>( \sigma )</th>
<th>10</th>
<th>15</th>
<th>25</th>
<th>35</th>
<th>50</th>
<th>65</th>
<th>75</th>
</tr>
</thead>
<tbody>
<tr>
<td>BM3D</td>
<td>33.31</td>
<td>31.10</td>
<td>28.57</td>
<td>27.08</td>
<td>25.62</td>
<td>24.68</td>
<td>24.20</td>
</tr>
<tr>
<td>MLP</td>
<td>33.50</td>
<td>–</td>
<td>28.97</td>
<td>27.48</td>
<td>26.02</td>
<td>25.10</td>
<td>24.58</td>
</tr>
<tr>
<td>TNRD</td>
<td>–</td>
<td>31.41</td>
<td>28.91</td>
<td>–</td>
<td>25.95</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>IRCNN</td>
<td>–</td>
<td><strong>31.63</strong></td>
<td><strong>29.14</strong></td>
<td>–</td>
<td><strong>26.17</strong></td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Ours</td>
<td><strong>33.58</strong></td>
<td><strong>31.44</strong></td>
<td><strong>29.05</strong></td>
<td><strong>27.56</strong></td>
<td><strong>26.06</strong></td>
<td><strong>25.12</strong></td>
<td><strong>24.61</strong></td>
</tr>
</tbody>
</table>

Fig. 3. Gaussian denoising performance profile relative to BM3D
on PASCAL. Image indices are sorted in ascending order of perfor-
mance gain relative to BM3D. Our improvement is demonstrated
by (i) decrease of the zero-crossing point, and (ii) consistently higher
values of gain. The comparison was made on 1K images for \( \sigma = 25 \).

with the ADAM optimizer [57] with a learning rate \( \alpha = 10^{-4} \),
\( \beta_1 = 0.9, \beta_2 = 0.999 \) and \( \epsilon = 10^{-8} \).

In the class-aware setup, the classifier was trained on noisy
grayscale images with the same noise statistics used at test
time after downsampling the image-resolution to a fixed size
of 128 \( \times \) 128 pixels. Downsampling the image allows faster
classification and better robustness to noise. The classifier was
trained for 10K mini-batches of 64 images using ADAM
optimizer with a learning rate of \( \alpha = 10^{-4} \), \( \beta_1 = 0.9, \beta_2 = 0.999 \) and \( \epsilon = 10^{-8} \), and the categorical cross-entropy
loss. Keep probability of 0.5 was used with dropout.

V. EXPERIMENTS

A. Class-agnostic denoising

This section evaluates the performance of our class-agnostic
method on Gaussian and Poisson noise using the following test
sets: (i) images from PASCAL VOC [58]; (ii) the commonly
used 68 test images chosen by [59] from the Berkeley segmen-
tation dataset [60]; and (iii) for Poisson noise we additionally
evaluated performance on the commonly used 10 images test
set as well as on a real low-light image. In all experiments
of this section, a separate network was trained for each noise
type and level using 8K images from PASCAL [58].

1) Gaussian denoising: We compare our network perfor-
mance at removing Gaussian noise with standard deviation
values between 1 and 75 against the following leading
methods: (i) BM3D [19]; (ii) multilayer perceptrons (MLP)
[7]; (iii) TNRD [9]; and (iv) IRCNN [1] using the pretrained
models provided online. For MLP, TNRD and IRCNN we used

\(^1\)Code available at github.com/TalRemez/deep_class_aware_denoising.
set as presented in Table V.

3) Darmstadt noise dataset: To test our network on real noisy data, we have evaluated its performance on the dataset presented in [62] using their official benchmark for sRGB images. For the evaluation, we denoised each of the RGB color channels separately (to make a fair comparison to BM3D, cross-color BM3D filtering (BM3D CFA) was applied on all channels together with its optimal parameters. The default MATLAB denoising and a Gray world white Blanching algorithms were used.

![Image 49x518 to 296x631]

Fig. 4. Poisson denoising performance profile relative to I+VST+BM3D on PASCAL. Image indices are sorted in ascending order of our performance.

<table>
<thead>
<tr>
<th>Peak</th>
<th>1</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>30</th>
</tr>
</thead>
<tbody>
<tr>
<td>I+VST+BM3D</td>
<td>22.71</td>
<td>23.70</td>
<td>24.78</td>
<td>26.08</td>
<td>25.85</td>
</tr>
<tr>
<td>IRCNN</td>
<td>22.70</td>
<td>23.98</td>
<td>25.22</td>
<td>26.58</td>
<td>29.44</td>
</tr>
<tr>
<td>Ours</td>
<td>22.87</td>
<td>24.09</td>
<td>25.36</td>
<td>26.70</td>
<td>29.56</td>
</tr>
</tbody>
</table>

TABLE III

POISSON DENOISING ON PASCAL. AVERAGE PSNR VALUES FOR DIFFERENT PEAK VALUES ON 1K TEST IMAGES AND 15 NOISE REALIZATIONS PER IMAGE.

![Table III]

![Table IV]

TABLE IV

POISSON DENOISING ON 68 IMAGE SET FROM [59]. AVERAGE PSNR VALUES OF 15 NOISE REALIZATIONS FOR DIFFERENT PEAK VALUES ON 68 IMAGES. RESULTS FOR NLSPCA, VST+BM3D and TRDPD reported in [8] were reproduced with the addition of the values for our method and for I+VST+BM3D.

![Table V]
though it has not been explicitly enforced at training, the error monotonically decreases with the layer depth (bottom left). This non-trivial behavior is consistently produced by the network on the vast majority of test images. These results align with the ones depicted in Fig. 6.

Fig. 8 presents failure cases of our class-agnostic method. It shows the five images with the worst performance compared to BM3D in the case of Gaussian noise with $\sigma = 25$. Interestingly, in all five cases, big portions of the image contain repeating patterns at a high spatial frequency, e.g., nets and fences. Clearly, in such cases collaborative filtering techniques such as BM3D are expected to perform particularly well.

### C. Class-aware denoising

As described in Section III-B, our class-aware denoiser comprises a classifier and a set of denoisers fine-tuned to the image classes produced by the classifier. Now we evaluate the boost in performance gained by fine-tuning a denoiser on a set of images belonging to a particular class.

For this experiment, we collected images from ImageNet [55] belonging to the following five classes: face, pet, flower, living room, and street. The 1500 images per class were split into train (60%), validation (20%) and test (20%) sets. We then trained a separate class-specific denoiser for each of the classes, initializing all weight to the values learned by our class-agnostic model and fine-tuning them to the specific class for 50K iterations. In addition, we trained a classifier to classify the noisy images as described in Section III-B.

### TABLE V

**POISSON DENOISING PERFORMANCE ON STANDARD IMAGES.** NUMERIC VALUES REPRESENT PSNR IN dB AVERAGED OVER FIVE NOISE REALIZATIONS. VALUES FOR PRIOR ART ALGORITHMS FOR PEAK VALUES OF 1 – 4 WERE TAKEN FROM [46]. FOR THE REST OF THE PEAK VALUES WE RAN THE CODE PUBLISHED BY THE AUTHORS; IN THE ABSENCE OF OPTIMAL PARAMETER SETTINGS, WE USED THOSE FOR PEAK = 4. TIMING VALUES PRESENTED ARE AVERAGES FOR IMAGES OF SIZE 256 x 256, FOR OUR METHOD WE PRESENT THE RUN-TIME ON TITAN-X GPU/INTEL E5-2630 2.20GHz CPU.

<table>
<thead>
<tr>
<th>Method</th>
<th>Peak</th>
<th>Flag</th>
<th>House</th>
<th>Cam</th>
<th>Man</th>
<th>Bridge</th>
<th>Saturn</th>
<th>Peppers</th>
<th>Boat</th>
<th>Couple</th>
<th>Hill</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLSPCA</td>
<td>20.15</td>
<td>24.26</td>
<td>20.97</td>
<td>22.93</td>
<td>20.21</td>
<td>27.99</td>
<td>21.07</td>
<td>22.49</td>
<td>22.33</td>
<td>23.51</td>
<td>123s</td>
<td>16s</td>
</tr>
<tr>
<td>NLSPCA bin</td>
<td>15.52</td>
<td>20.94</td>
<td>18.27</td>
<td>19.98</td>
<td>18.32</td>
<td>22.02</td>
<td>17.72</td>
<td>20.29</td>
<td>20.25</td>
<td>20.99</td>
<td>13s</td>
<td>9.8s</td>
</tr>
<tr>
<td>SPDA</td>
<td>25.76</td>
<td>25.23</td>
<td>21.72</td>
<td>-</td>
<td>20.53</td>
<td>31.13</td>
<td>22.2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>8h</td>
<td>25min</td>
</tr>
<tr>
<td>SPDA bin</td>
<td>19.42</td>
<td>22.07</td>
<td>19.95</td>
<td>22.18</td>
<td>19.26</td>
<td>29.71</td>
<td>20.19</td>
<td>21.76</td>
<td>21.69</td>
<td>22.82</td>
<td>31min</td>
<td>10.4s</td>
</tr>
<tr>
<td>P^IP</td>
<td>22.49</td>
<td>26.33</td>
<td>23.29</td>
<td>24.61</td>
<td>21.11</td>
<td>30.82</td>
<td>23.88</td>
<td>24.10</td>
<td>23.99</td>
<td>25.28</td>
<td>7.4s</td>
<td>0.8s</td>
</tr>
<tr>
<td>VST+BM3D</td>
<td>22.93</td>
<td>25.49</td>
<td>23.28</td>
<td>24.32</td>
<td>21.51</td>
<td>30.82</td>
<td>23.88</td>
<td>24.10</td>
<td>23.99</td>
<td>25.28</td>
<td>7.4s</td>
<td>0.8s</td>
</tr>
<tr>
<td>VST+BM3D bin</td>
<td>20.43</td>
<td>24.40</td>
<td>23.39</td>
<td>21.45</td>
<td>40.03</td>
<td>23.73</td>
<td>23.92</td>
<td>23.84</td>
<td>24.69</td>
<td>10.0s</td>
<td>1.4s</td>
<td>0.4s</td>
</tr>
<tr>
<td>Ours</td>
<td>23.18</td>
<td>26.59</td>
<td>24.77</td>
<td>24.77</td>
<td>21.81</td>
<td>30.02</td>
<td>24.83</td>
<td>24.86</td>
<td>24.60</td>
<td>25.01</td>
<td>0.4s</td>
<td>1.3s</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Peak</th>
<th>Flag</th>
<th>House</th>
<th>Cam</th>
<th>Man</th>
<th>Bridge</th>
<th>Saturn</th>
<th>Peppers</th>
<th>Boat</th>
<th>Couple</th>
<th>Hill</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLSPCA</td>
<td>14.87</td>
<td>20.87</td>
<td>18.21</td>
<td>19.76</td>
<td>18.23</td>
<td>21.44</td>
<td>17.67</td>
<td>20.20</td>
<td>20.21</td>
<td>20.93</td>
<td>60s</td>
<td>9.8s</td>
</tr>
<tr>
<td>NLSPCA bin</td>
<td>26.85</td>
<td>26.30</td>
<td>22.24</td>
<td>24.36</td>
<td>21.05</td>
<td>32.39</td>
<td>22.89</td>
<td>23.50</td>
<td>23.37</td>
<td>24.93</td>
<td>9.8s</td>
<td>25min</td>
</tr>
<tr>
<td>SPDA</td>
<td>23.10</td>
<td>27.36</td>
<td>24.97</td>
<td>24.96</td>
<td>21.68</td>
<td>32.88</td>
<td>24.94</td>
<td>25.03</td>
<td>25.06</td>
<td>24.50</td>
<td>167s</td>
<td>10.4s</td>
</tr>
<tr>
<td>P^IP</td>
<td>25.54</td>
<td>27.95</td>
<td>25.74</td>
<td>25.81</td>
<td>22.72</td>
<td>34.52</td>
<td>23.52</td>
<td>25.79</td>
<td>26.06</td>
<td>5.1s</td>
<td>0.4s</td>
<td>1.3s</td>
</tr>
<tr>
<td>VST+BM3D</td>
<td>25.73</td>
<td>28.42</td>
<td>26.35</td>
<td>26.10</td>
<td>22.91</td>
<td>32.28</td>
<td>24.65</td>
<td>26.33</td>
<td>26.23</td>
<td>26.26</td>
<td>25.01</td>
<td>3.5s</td>
</tr>
<tr>
<td>VST+BM3D bin</td>
<td>27.10</td>
<td>27.06</td>
<td>22.47</td>
<td>25.02</td>
<td>21.22</td>
<td>35.08</td>
<td>23.61</td>
<td>24.55</td>
<td>24.06</td>
<td>25.88</td>
<td>0.4s</td>
<td>1.3s</td>
</tr>
<tr>
<td>Ours</td>
<td>28.94</td>
<td>31.67</td>
<td>29.21</td>
<td>28.74</td>
<td>25.42</td>
<td>36.20</td>
<td>29.77</td>
<td>29.06</td>
<td>29.13</td>
<td>28.71</td>
<td>0.4s</td>
<td>1.3s</td>
</tr>
</tbody>
</table>

### TABLE VI

**DARMSTADT NOISE DATASET.** AVERAGE PSNR AND SSIM.

<table>
<thead>
<tr>
<th>Method</th>
<th>BM3D</th>
<th>MLP</th>
<th>DnCNN</th>
<th>TNRD</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR</td>
<td>34.51</td>
<td>34.23</td>
<td>32.43</td>
<td>33.05</td>
<td>35.08</td>
</tr>
<tr>
<td>SSIM</td>
<td>0.85</td>
<td>0.83</td>
<td>0.79</td>
<td>0.83</td>
<td>0.87</td>
</tr>
</tbody>
</table>
We trained the classifier and the denoisers for two separate scenarios: (i) additive Gaussian noise with $\sigma = 25$; and (ii) Poisson noise with peak value 8. We then compared on the test sets the performance of (a) our class specific denoisers using oracle class labels; (b) a full pipeline system that uses the learned classifier to determine the class of each image and applies the denoiser that belongs to that class; (c) our class-agnostic model trained on PASCAL; and (d) previous denoising methods using their available code and/or pretrained models. To evaluate the ability of other class-agnostic learning based methods to be transformed into class-aware ones as well as to have a fair comparison, we transformed IRCNN from being class-agnostic into class-aware using the exact same framework as ours (i.e., having specialized denoisers for each class and a classifier to select the image type). To this end we used our reimplementation of IRCNN and trained it using exactly the same regime as the one used to train our networks.

Figure 9 presents PSNR values for Gaussian and Poisson noise. It is evident that the class-specific models boost performance over their class-agnostic counterpart as well as other algorithms; this is true both for our method and for IRCNN. Note also that using the full pipeline that includes a classifier exhibits similar behavior to the oracle. It is also evident that in almost all cases our method outperforms both the class-agnostic and the class-aware versions of IRCNN. Table VII presents the classifiers performance on the noisy test sets.

**Cross class denoising.** To further demonstrate the effect of refining a denoiser to a particular class, we tested each class-specific denoiser on images from other classes. The outcome of mismatching the image and denoiser classes is evident both qualitatively and quantitatively.

The top row of Fig. 10 presents a comparison of denoisers fine-tuned to the *street* and *face* classes applied to a noisy face image. The street-specific denoiser produces noticeable artifacts around the eye, cheek and hair areas. Moreover, the edges appear too sharp and seem to favor horizontal and vertical edges. This is not very surprising as street images contain mainly man-made rectangle-shaped structures. In the second row, strong artifacts appear on the hamster’s fur when the image is processed by the *living room*-specific denoiser, while the *pet*-specific denoiser produces a more naturally looking result. Examples on the canonical images *House* and *Lena* are presented in the bottom two rows. Notice how the *street*-specific denoiser reconstructs sharp boundaries of the building whereas the *face*-specific counterpart smears them.

To quantify the effect of class mismatch, we evaluate the percentage of wins of every fine-tuned denoiser on each type of image class. A ‘win’ means that a particular denoiser produces the highest PSNR among all the others. Fig. 11 presents a matrix with all the combinations of class-specific denoisers and image classes.

**D. Video denoising**

This section demonstrates how our method can be extended to video denoising. To this end, we propose to alter our network architecture described in Fig. 2. The main difference between our image and video architectures lies in how we provide the inputs to the network and what is the output we wish to obtain. As opposed to image denoising, where we feed a single gray-scale image as input, for videos we feed a sequence of $T$ stacked gray-scale images. We request the network to estimate the noise that should be subtracted from the central image of the input sequence. In practice, this only implies an increase in the depth of the kernels at the first layer of the network from 1 to $T$ and is therefore relatively inexpensive. When applying our model to RGB videos, we denoise each color channel independently.

To demonstrate the efficacy of our method for high-resolution videos contaminated by Poisson noise with peak values of 8 and 30 (independent for each color channel), we compared our architecture with 1, 5, and 11 frame-long sequences, using the following leading methods as a reference: I+VST+BM3D [46] and VST+VBM4D [64], [65].

Our training set was comprised of 17 YouTube videos $1920 \times 1080$ with varying content and length (between 4
Fig. 6. **Gradual denoising process.** The top row presents the noisy image (left) and the intermediate result obtained by removing the noise estimated up to the respective layer depth. The second row presents the ground truth image (left) and the noise estimates produced by individual layers; noise estimation images have been scaled for display purposes. Zoom-in for a better view of the fine details and noise. Images have been contaminated by Gaussian noise with $\sigma = 25$. PSNR values are presented below each image.

and 10 minuted long) comprising 161K training frames. All algorithms were tested on 6 unseen videos, 2000 frames from each video (frames 20-2020 to give all temporal algorithms a warm-up period of 20 frames), 12000 frames in total. The full list of videos used in our experiments appears in Appendix A.

I+VST+BM3D was applied to each image color channel separately. Due to the heavy memory usage of VBM4D, while testing this method, videos were sliced to blocks of 30 frames with size $384 \times 270$, which were stitched back after being denoised. Moreover, as this method was designed for Gaussian noise, the Anscombe transform was used to whiten the noise.

Tables VIII and IX present quantitative results, in which a breakdown of PSNR performance is given by video. It is evident that our proposed technique outperforms the existing techniques for video denoising. As expected, our method improves when more frames are used for denoising.

Figs. 12 and 13 present some qualitative examples. The time required per average frame are 450 seconds for I+VST+BM3D, 97 seconds for VBM3D, 3, 3.9, and 4.8 seconds for our network with $T = 1$, 5, and 11 receptively. Training was done in a similar fashion to our image models while using sequences of length $T$ with random patches of size $128 \times 128$. We use Adam optimizer with a learning rate of $1e^{-4}$, patches are randomly flipped horizontally and the batch size is 64. The models with $T=1$, 5, and 11 converged after 130K,170K, and 330K iterations respectively.

E. Ablation study

In the following experiments, we evaluated the effect of various hyper-parameters of our network on its performance. In all cases, we trained on the PASCAL training set with an additive Gaussian noise with $\sigma = 25$ using the same methodology described in Section IV. Testing was performed on the 1000 test images of PASCAL.
processing tasks such as blind deblurring [66] and burst denoising [61] also reported no advantage in the use of BN.

b) Skip connections: We evaluated the contribution of the intermediate skip connection at each layer (the dark blue blocks in Fig. 2). To this end we trained two networks that are identical except that one has a skip connection at each layer (as in Fig. 2), while the other have no skip connections at all (has 64 feedforward filters at each layer). Both networks had 20 layers. The network without the skip connections achieved a PSNR of 30.26 dB, while the one with the proposed skip connections achieved 30.33 dB.

c) Network depth: We tested the performance of our network for different depths ranging from 5 layers up to 40. In all cases the networks had 64 kernels per layer and intermediate skip connections. The results are presented in Table X. We selected a depth of 20 layers as beyond it, the network has diminishing returns. With 20 layers we achieve a similar performance to a 40 layers network but with half the number of trainable parameters, memory and inference time.

d) Number of kernels: We tested the effect of the number kernels at each network layer for a network with 20 layers with skip connections. The values tested were 16, 32, 64, and 96 kernels per layer. The results are presented in Table XI. We selected to use 64 kernels per layer as this reduces by half the trainable parameters compared to having 96 kernels per layer and in addition we saw that adding more kernels usually resulted in poorer generalization of the network on other datasets such as BSD (when we did not train on those datasets but rather only tested on them). Interestingly, the authors of IRCNN found that the same number of kernels led to the best performance.

VI. DISCUSSION

We introduced a novel fully convolutional neural network for Gaussian and Poisson image denoising with performance often exceeding the state-of-the-art. Our network architecture is inherently transparent, in the sense that all intermediate layers are extracted and directly contribute to the noise estimation. This is helpful in overcoming the “black-box” nature of neural
Ground Truth

BM3D

Ours

<table>
<thead>
<tr>
<th>kernels</th>
<th>16</th>
<th>32</th>
<th>64</th>
<th>96</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR</td>
<td>29.84</td>
<td>30.17</td>
<td>30.33</td>
<td>30.40</td>
</tr>
</tbody>
</table>

**Fig. 8.** Failure examples. Examples of failures of our method compared to BM3D, we selected the five worst images from the PASCAL test set (for Gaussian noise $\sigma = 25$). Interestingly all of these imaged have repeated structures, such as nets or fences.

**Table XI**

**NUMBER OF KERNELS PER LAYER. AVERAGE PSNR VALUES ON 1K PASCAL IMAGE TEST-SET FOR DIFFERENT NUMBERS OF KERNELS PER NETWORK LAYER.**

networks as it allows us to gain several interesting insights on the denoising process.

In addition, we showed that using a classifier to route the input image to a class-specific network is preferable to a universal filter, achieving an additional boost of up to 0.4 dB PSNR over a class-agnostic denoiser. That said, the decision to split according to a global semantic class as done in this paper was made due to the immediate availability of labeled data and off-the-shelf classifiers that are relatively resilient to noise. Yet, this splitting scheme may be sub-optimal and other choices for data partitioning could be made. In particular, the splitting scheme could be learned automatically by incorporating it into the network architecture. It would then lose its simple semantic interpretation, and instead yield abstract classes, perhaps varying across different regions of the image.

**Limitations.** In most denoising methods, some set of parameters is adjusted according to the noise type and level. While in algorithms such as BM3D this set is small and requires a small amount of memory, in deep learning methods like MLP, TNRD, IRCNN and ours, it is much larger. Furthermore, in our class aware denoising framework, it grows linearly with the number of classes. Clearly, being specific to one class deteriorates the performance to the other ones as shown in Fig. 10 and therefore a denoiser is required per each class. Integrating the splitting scheme into the network might mitigate this issue. Note also that when a class specific denoiser is trained, one should verify that the variety of images used for training is large and diverse enough to avoid overfitting.

Another limitation is a decrease in the performance gain of our method with the decrease of the peak value (i.e., in the presence of stronger noise). A possible explanation for this phenomenon might lie in the fact that extremely noisy images no longer resemble natural ones and small convolution kernels have a harder time learning such patterns. A different architecture might be more adequate for such scenarios.

An additional limitation we leave to be solved in future work is the ability to handle the noise in real images (see for example [62]), where the noise is of a mixed type and its level cannot be assumed to be known prior to the denoising process.

**APPENDIX A**

**LIST OF VIDEOS**

The list of videos used for training, validation, and testing of the proposed video denoising network is provided below.

**A. Video training set**

1) Extreme Bungy Jumping with Cliff Jump Shenanigans! 4K! youtu.be/l9m4cW2yxy0
2) Motorized Drift Trike and Blokart in 4K! youtu.be/Mf2wAEYtrlo
3) Robotic Dolphin and Flying Water Car - In 4K! youtu.be/dkpzgS9rdTk
4) World’s Best Basketball Freestyle Dunks - 4k youtu.be/5rd_WwD_aOQ
5) Hoverboard in Real Life! In 4K! youtu.be/gMaDhkJNA2g
6) Barefoot Skiing behind Airplane in 4K - Insane! youtu.be/vdTrr_VRKgU
Fig. 9. **Class-aware denoising on ImageNet.** Average PSNR values on images belonging to five semantic classes for Gaussian noise with $\sigma = 25$ (top) and Poisson noise with peak value $= 8$ (bottom).

7) Assassin’s Creed Unity Meets Parkour in Real Life - 4K! youtu.be/S8b1zWOgOKA
8) Top 5 moments in Roland Garros - Best rallies youtu.be/a7O54L_vcyo
9) The Nexus interrupt the main event and reap destruction Raw youtu.be/vVViqoqzgNw
10) May Win Compilation 2014 MW youtu.be/UgY9eR3Zd0k
11) Insane Human Skeeball! youtu.be/nyMgJ3z8U9I
12) GoPro Sketchy Cornice Rappel youtu.be/7fGGEyY3qVk
13) Dubai in 4K - City of Gold youtu.be/SLaYPmhse30
14) Sky Racers! - Dubai! 4K! youtu.be/eCTQvyXELaFI
15) GoPro HD HERO camera Base Jump Movie youtu.be/mRzhBkZNQFI
16) GoPro Combing Valparaiso’s Hills youtu.be/cN-YTeSnE6c
17) GoPro Moab Towers & Magic Backpacks youtu.be/fVcV9ItdZ8w

B. **Video validation set**

1) DevinSupertramp - Best of 2014 youtu.be/bpz-Pq61DLw
2) 41-Man Battle Royal for a Championship Match youtu.be/n3PswaGlZt4

---

Fig. 10. **Cross-class denoising.** Examples of using correct (middle) and mismatched (right) class-specific denoisers on images with Gaussian noise, $\sigma = 25$. PSNR values and denoiser type are listed above each denoised image. The reader is encouraged to zoom in for a better view of the artifacts.

Fig. 11. **Cross class denoising** for Gaussian noise with $\sigma = 25$ (left) and Poisson noise with peak value $8$ (right). The rows represent the tested semantic class of images while the columns the class-aware denoisers used. The $(i, j)$-th element in the matrix shows the probability of the $j$-th class-aware denoiser to outperform all other class-aware denoisers on the $i$-th class of images.

C. **Video test set**

1) Sandboarding Supertramp Style - 4K! youtu.be/0ENviLq-Xfl
2) World’s Craziest Teeterboard Flips - Streaks Show in 4K! youtu.be/95f4kb5XR3U
3) John Cena & Randy Orton battle the entire Raw roster Raw youtu.be/ndgVcE7w0uo
4) GoPro Rocky Cliff Huck In The French Alps youtu.be/1EOHQQBZbJw
5) GoPro The Untold Story of Ryan Villopoto youtu.be/4oJc8If2Gpc
<table>
<thead>
<tr>
<th>Ground truth</th>
<th>I+VST+BM3D</th>
<th>vBM4D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours (1)</td>
<td>Ours (5)</td>
<td>Ours (11)</td>
</tr>
</tbody>
</table>

Fig. 12. **Poisson Video Denoising Example.** Denoising comparison of a low-light video with a peak value 8. Intensities were scaled for display purposes.
Fig. 13. **Poisson Video Denoising Example.** Denoising comparison of a low-light video with a peak value 8. Intensities were scaled for display purposes.
<table>
<thead>
<tr>
<th>Image</th>
<th>Ground Truth</th>
<th>Noisy</th>
<th>I+VST+BM3D</th>
<th>IRCNN</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>24.56 dB</td>
<td>25.62 dB</td>
<td>25.83 dB</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>24.76 dB</td>
<td>25.78 dB</td>
<td>26.00 dB</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>23.40 dB</td>
<td>24.47 dB</td>
<td>24.66 dB</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>30.14 dB</td>
<td>31.43 dB</td>
<td>31.61 dB</td>
</tr>
</tbody>
</table>

Fig. 14. Poisson denoising examples from PASCAL. For Poisson noise with peak 8, PSNR values appear below each of the images.


### REFERENCES

21. O. Bryt and M. Elad, “Compression of facial images using the K-SVD


[46] L. Azzari and A. Foi, “Variance stabilization for noisy-estimate com-

Fig. 15. Gaussian (σ = 25) denoising examples from ImageNet. We present four class-agnostic methods and two class-aware ones. PSNR values appear below each of the images.
Fig. 16. Poisson (peak= 8) denoising examples from ImageNet. PSNR values appear below each of the images.