Real-World Mobile Image Denoising Dataset with Efficient Baselines

Roman Flepp[1] Andrey Ignatov[1,3] Radu Timofte[1,2,3] Luc Van Gool[1,3]

[1] Computer Vision Laboratory, ETH Zürich [2] Computer Vision Laboratory, University of Würzburg [3] AI Witchlabs Ltd., Zollikerberg

r.flepp@hotmail.com, {andrey, timofter, vangool}@vision.ee.ethz.ch

Abstract

The recently increased role of mobile photography has raised the standards of on-device photo processing tremendously. Despite the latest advancements in camera hardware, the mobile camera sensor area cannot be increased significantly due to physical constraints, leading to a pixel size of 0.6–2.0 μm, which results in strong image noise even in moderate lighting conditions. In the era of deep learning, one can train a CNN model to perform robust image denoising. However, there is still a lack of a substantially diverse dataset for this task. To address this problem, we introduce a novel Mobile Image Denoising Dataset (MIDD) comprising over 400,000 noisy / noise-free image pairs captured under various conditions by 20 different mobile camera sensors. Additionally, we propose a new DPreview test set consisting of data from 294 different cameras for precise model evaluation. Furthermore, we present the efficient baseline model SplitterNet for the considered mobile image denoising task that achieves high numerical and visual results, while being able to process 8MP photos directly on smartphone GPUs in under one second. Thereby outperforming models with similar runtimes. This model is also compatible with recent mobile NPUs, demonstrating an even higher speed when deployed on them. The conducted experiments demonstrate high robustness of the proposed solution when applied to images from previously unseen sensors, showing its high generalizability. The datasets, code and models can be found on the official project website[1,2].

1. Introduction

In 2024, there are more than 7 billion smartphone users [4], thus making mobile photography, and especially mobile image denoising, an important research area. The smaller the sensor, the less amount of light can be captured per pixel, which in turn leads to elevated ISO levels or longer exposure times for the image to have the correct exposure settings. In DSLR photography, it is more common to use longer exposure times to use lower ISO settings and effectively reduce the noise in the image. To keep the camera and its sensor stable during enlarged periods of exposure time, a tripod and a remote camera controller are mainly used. Longer exposure times are often not practical in mobile photography, because of moving subjects or movement of the camera sensor while still not offering visually pleasing results [16]. Hence, higher ISO levels are more desirable when it comes to practicality. Those elevated ISO levels in turn lead to increased noise in the captured image. Thus, performing noise reduction on mobile devices is a crucial factor in producing a visually pleasing output image, and image denoising "on the go", without spending extended effort to capture a single image, is the goal.

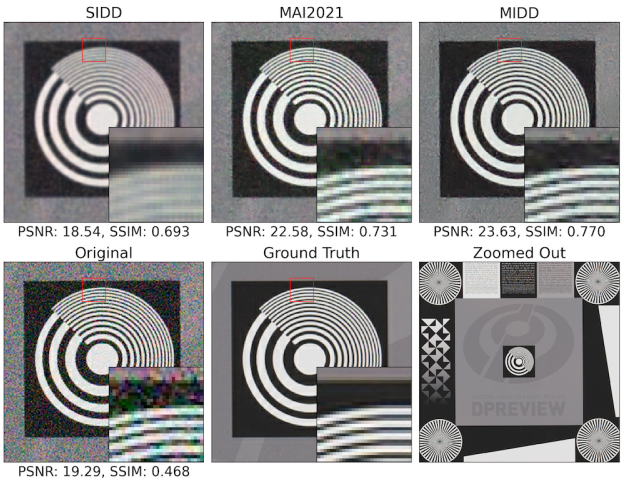

Figure 1. Visual comparison of denoising results obtained with the SplitterNet model trained on different datasets using the same training pipeline. The image is part of the DPreview [2] test set and is independent of all used training datasets, thus offering an optimal comparison ground. It can be clearly seen that the SplitterNet trained on the SIDD [9] introduces a strong blur in the denoised image, while the other denoised images show less blur, and the model trained on the MIDD offers the best denoising capabilities.

Different classical approaches before the era of deep learning have been proposed in the past [10, 11, 13, 22]. Those have been quantitatively outperformed since a few years by convolutional neural networks-based deep learn-

[1] https://people.ee.ethz.ch/~ihnatova/midd.html

[2] https://github.com/rflepp/Efficient_Mobile_Denoising_Models

Figure 2. Sample training image pair: the original noisy photo on the left and the corresponding noise-free ground truth image on the right.

ing algorithms that have shown strong denoising capabilities [14, 25, 26] also for mobile phones [8, 16, 21]. Image denoising using CNNs on mobile devices comes with the additional challenge of limited computing power. Mobile GPUs and TPUs have made big advancements in terms of performance over the last couple of years, exceeding the demands of most users. Even though these advancements have been astonishing, it is still important for deep learning denoising models to be as efficient as possible during inference to fulfill the goal of "on-the-go" image denoising.

Recent works have focused not only on better visual performances but also on faster inference times as seen in the MAI2021 challenge for mobile image denoising [16] or in the current state-of-the-art model KBNet [26] that offers a better PSNR and faster inference time over comparable networks as NAFNet [12]. State-of-the-art networks have used diverse attention mechanisms such as Simplified Channel Attention (SCA) in NAFNet, Kernel Basis Attention (KBA) in KBNet or Spatial Multi-head Self-Attention (Spatial MSA) and Channel Multi-head Self-Attention (Channel MSA) in the winning model of the NTIRE 2023 challenge in image denoising by *Apply_AI* [21]. These attention mechanisms offer the ability for the network to focus on aspects of spatial information (KBA, Spatial MSA) or channel information (SCA, Channel MSA) and aggregate those in a meaningful manner.

To get performant and robust real-world models and results, it is not only important to have a great image-denoising model but also an extensive collection of real-world mobile image data for training. The latter part is often neglected, as there have not been many large real-world mobile image-denoising datasets proposed. Currently, the SIDD [9] and DND [23] are the broadly used datasets for mobile image denoising training and performance assessment. SIDD consists of about 30,000 images. There are 150 images taken for each of the 5 camera sensors in 10 different static indoor scenes that are illuminated in 4 different conditions. This dataset offers a large body of images but is limited to only 10 indoor scenes and only 5 camera sensors. DND consists of 50 low ISO to high ISO image pairs of different scenarios, but 50 images is not enough for image denoising model training to learn meaningful denoising capabilities. Thus, there is a strong need for a large real-world image denoising dataset including a broad range of real-world scenes ranging over a variety of lighting conditions and camera sensors.

Contributions This work introduces MIDD, a large mobile image denoising dataset consisting of 400,000 noisy photos and 20,000 corresponding ground truth images captured in real-world scenarios including indoor and outdoor scenes, artificial and natural lighting, bright and dark scenes. During data collection, 20 camera sensors from recent mobile phone models were used, resulting in different noise patterns. When employing Convolutional Neural Network (CNN)-based techniques, a substantial improvement is observed by utilizing our data compared to previous options — SIDD or MAI2021 datasets. Additionally, we present an extensive DPreview test dataset that includes photos from 294 cameras and offers data to analyze the robustness of a model or the quality of a dataset. To give a robust numerical baseline, we introduce SplitterNet, a new model that is jointly optimized for PSNR, SSIM, and inference performance and outperforms the winner of the MAI2021 challenge in all three metrics, hence offering a robust baseline for the mobile image denoising task. The model is evaluated on different mobile devices, showing the real-world applicability of our solution. The SplitterNet seamlessly integrates with the TensorFlow Lite framework [6]. Its deployments are versatile, as the model can run smoothly on any mobile device with AI acceleration capabilities that utilize either the Android Neural Networks API (NNAPI) [5] or custom TensorFlow Lite delegates [7].

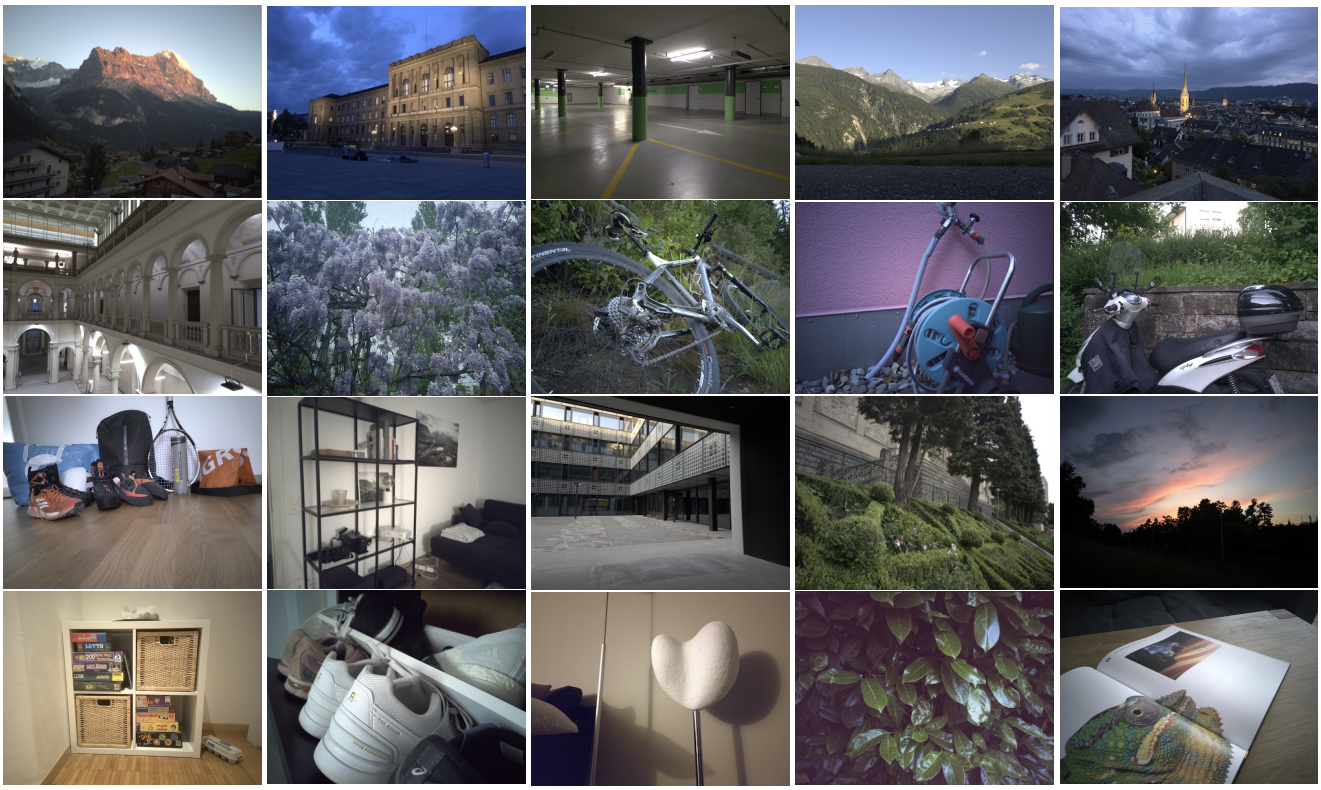

Figure 3. Sample images from the MIDD dataset showing the diversity in scenes, lighting conditions, and consequently noise distributions.

2. Dataset

In this chapter, we present the capturing procedure and the ground truth estimation process used to create the Mobile Image Denoising Dataset (MIDD) as well as the DPreview dataset.

2.1. Dataset Collection

Our main goal was to develop a diverse and comprehensive real image denoising dataset consisting of photos captured using various mobile camera sensors. To achieve this, approximately 20,000 noisy images were collected for each of 20 different mobile camera sensors. The burst mode was used to capture 20 RAW Bayer images for each static scene with a custom Android application utilizing the Camera2 API [1], which allowed to bypass the phone's image signal processor (ISP) and prevent any internal image denoising. All photos were taken using a tripod and a remote control, ensuring no movement of the camera or its sensor. This approach yielded over 200K different noisy-to-ground-truth image pairs for each sensor, with 20 pairs per scene.

About 20% of the dataset was captured during the daytime, 50%–60% under less ideal conditions (like blue hour, dark indoor areas, artificial lighting), and roughly 20% during night-time or in very low light conditions. These varied conditions were crucial to encompass a broad range of real-world lighting situations as they have a strong impact on the noise distribution. Utilizing 20 different sensors, each with a unique noise pattern, was essential to introduce an even more diverse noise distribution. This process resulted in over 400,000 images in total (20 images per burst $\times$ 1000 scenes $\times$ 20 sensors).

The selected scenes varied, including both indoor and outdoor settings. The range of scenes spanned from office settings to mountain landscapes, as illustrated in Fig. 2, thus aligning to create a versatile and broad real-world image-denoising dataset.

To ensure high-quality images with minimal movement of objects or camera sensors, each picture was manually checked by a human annotator. Images displaying blur or misalignment due to the optical image stabilization were removed from the dataset.

2.2. Sensors

To introduce a large dataset noise distribution diversity, a wide range of mobile camera sensors from various manufacturers and generations is used. Different camera sensors add unique noise distributions to the output images, allowing machine learning models to learn and adapt to these variations, enhancing their real-world applicability and robustness. Table 1 provides details on the used sensors. The

Sensor	Sensor Resolution	Pixel Size	Year	Sensor	Type	Phone
Sony IMX179	8 MP (3288×2512)	1.4 μm	2017	RGB	Front	Google Pixel 2
Sony IMX258	13 MP (4224×3136)	1.12 μm	2015	RGB / Monochrome	Main	Essential Phone
Sony IMX268	8 MP (3840×2160)	1.12 μm	2016	RGB	Front	Essential Phone
Sony IMX351	16MP (4656×3496)	1.0 μm	2018	RGB	Main	LG G7 ThinQ
Sony IMX362	12 MP (4000×3000)	1.4 μm	2017	RGB	Main	Google Pixel 2
Sony IMX476	20 MP (5184×3888)	1.0 μm	2019	RGB	Front	Nokia 9
Sony IMX586	48 MP (8000×6000)	0.8 μm	2018	RGB / Quad Bayer	Main	Honor View 20
Sony IMX686	64 MP (9248×6944)	0.8 μm	2019	RGB / Quad Bayer	Main	Realme X7 Pro
Sony IMX766	50MP (8192 × 6144)	1.0 μm	2020	RGB / Quad Bayer	Main	Snapdragon Phone
ISOCELL 3J1	10 MP (3648×2736)	1.22 μm	2019	RGB	Front	Google Pixel 6
ISOCELL 2L4	12 MP (4032×3024)	1.4 μm	2019	RGB / Bayer	Main	Samsung Galaxy S10
ISOCELL 3P9	16 MP (4608×3456)	1.0 μm	2018	RGB / Tetracell	Front	Oppo A92s
ISOCELL 3T2	20 MP (5184×3880)	0.80 μm	2020	RGB / Dual Tetrapixel	Front	Redmi K30 Ultra
ISOCELL 3M5	50 MP (8160×6144)	0.8 μm	2021	RGB / Bayer	Main	Xiaomi Mi 9
ISOCELL GN1	50 MP (8160×6144)	1.2 μm	2020	RGB / Dual Tetrapixel	Main	Google Pixel 6
ISOCELL HM3	108 MP (12000×9000)	1.0 μm	2021	Nonapixel RGB Bayer	Main	Samsung Galaxy S22 Ultra
OmniVision OV32A	32MP (6560×5480)	0.8 μm	2019	RGB	Front	Realme X7 Pro
OmniVision OV48B	48 MP (8000×6000)	0.8 μm	2019	RGB / Quad Bayer	Main	Oppo A92s
OmniVision OV64B	64MP (9248 × 6944)	0.7 μm	2020	RGB	Front	Snapdragon Phone
Hynix SL846	8 MP (3264×2448)	1.125 μm	-	RGB	Front	LG G7 ThinQ

Table 1. Camera sensors used in the proposed MIDD dataset, their physical characteristics, and the actual device used for capturing photos.

dataset contains popular mobile sensors manufactured by *Sony*, *ISOCELL*, *OmniVision* and *Hynix* with the resolution ranging from 8 to 108 megapixels. It is important to note that the resolution of the final images might differ from the physical sensor resolution due to internal pixel bucketing, a technique widely used nowadays in many recent cameras. The dataset encompasses sensors produced from 2015 to 2021. This broad spectrum of sensor technologies contributes to the dataset's value, ultimately improving the performance of models trained on it in image-denoising tasks.

2.3. RGB Image Retrieval

Captured RAW photos were converted to RGB format using a minimal Image Signal Processor (ISP) incorporated in the RawPy wrapper for LibRaw [3]. The required demosaicing process for the Bayer pattern on the CMOS sensor was handled by RawPy, which also transformed the image into the AdobeRGB color spectrum. The final RGB images were saved as PNG files without any additional image enhancements. Using this technique, we ensure that no internal image enhancement (including image denoising) was performed by mobile devices. A noted concern was the presence of defective pixels, which appeared as blown-out values in both noisy and denoised images. However, as these defective pixels are consistent across the training and ground truth images, they should not have any noticeable effect on the learning process.

2.4. Ground Truth Estimation

To get the ground truth noise-free images, we used 20 burst photos per scene to calculate the mean image as shown in Eq. (1). Here, $\mathbf{G}_{i,j}$ is the pixel value at position (i, j) of the ground truth image, and $\mathbf{I}_{l,i,j}$ is the pixel value at position (i, j) of the l-th noisy image:

$$G_{i,j} = \frac{1}{20} \sum_{l=1}^{20} I_{l,i,j} \tag{1}$$

By averaging over 20 noisy images, we expect to get a noise-free pixel and therefore a mostly noise-free image. As noise is a random process, this method has been shown to perform well for ground truth estimation [9, 16]. As a consequence of our static image-capturing technique, neither alignment nor further image processing was needed. The dataset's integrity was challenged by scene motion, potentially causing motion blur, especially in windy conditions or with moving objects. To mitigate this, images were preferably captured in calm conditions. Small instances of blur were considered acceptable, as the extensive size of the MIDD dataset allows deep learning models to view these as outliers.

2.5. Dataset Format

The Mobile Image Denoising Dataset (MIDD) includes data for denoising RGB and RAW images, all stored as PNG and DNG files, respectively. Unlike the SIDD dataset, MIDD is available in its full size, reserving around 25% of data for evaluation and benchmarking. With multiple noisy images provided for each ground truth, the dataset supports training burst image denoising models, which process multiple noisy inputs to produce a superior denoised output.

2.6. DPreview Test Set

We also introduce an additional dataset collected by web-scraping the DPreview website [2]. This site features

Figure 4. A sample image from the newly proposed DPreview [2] dataset. The presented scene is optimized for camera quality analysis and offers a wide range of colors, textures, and details. The dataset consists of over 294 sensors that capture this scene.

an extensive camera comparison tool that showcases photos captured with various ISO levels for 294 different camera sensors. These range from low-end digital pocket cameras and phones to high-end cameras, including the Phase One IQ4 with its 150MP image sensor. An example photo from this dataset is shown in Fig. 4. The methodology for creating the images was consistent across all devices. A camera or phone was positioned centrally above the same scene, capturing RAW images starting from the lowest ISO level and ascending through the ISO range as far as each camera allowed. The chosen scene provides a rich array of colors, details, and intricate structures, making it an ideal environment for photography analysis.

The alignment of the collected image data was performed using a sliding window approach. The images were then batched, with the lowest ISO image serving as the ground truth and the highest ISO levels as noisy samples. What sets this dataset apart is the unprecedented variety of image sensors it incorporates, surpassing in this aspect all previously used image denoising datasets by at least an order of magnitude. Consequently, this dataset provides an excellent resource for evaluating and testing image denoising models, as well as for assessing image denoising datasets themselves as will be discussed later.

3. Efficient Mobile Image Denoising Baseline

To offer a robust baseline for the mobile image denoising task focused on optimizing the visual, numerical, and runtime performance, we introduce the SplitterNet model.

3.1. Model Architecture

The SplitterNet architecture was developed taking into account various constraints imposed by mobile AI accelerators such as smartphone NPUs and GPUs. To reduce the model complexity, it performs splitting of tensors throughout the network to run the convolution operation on a smaller split tensor versus convolving the tensor without splitting. The split operation has also been shown to perform well in the MAI2021 [16] challenge. From a high-level view, the SplitterNet is a U-Net-based model with multiple paths, its overall architecture is shown in Fig. 5. The workflow of the model is as follows. After convolving the input, it is split along the channel axis. The resulting tensor is then convolved again while reducing the width and height dimensions of the tensor and increasing the channel size to the initial channel number. Then, the next split operation is performed and the outputs thereof are convolved again. This procedure is repeated for each encoding step. In the lowest levels of the architecture, where the image information has been efficiently extracted by the encoding path, a combination of spatial and channel attention is used. In this middle block, attention mechanisms are used to extract the most important features (Fig. 5, right). There are as many parallel middle blocks in the architecture as there are encoding and decoding levels. A decoding block with transposed convolution and skip connection layers is used next. In the transposed convolution layer, the channel number is reduced while the spatial dimensions are extended. After each transposed convolution, different branches get concatenated again. In the last step, the channel size is reduced back to 3 and the denoised RGB image is returned.

3.2. Technical details

The model was implemented in TensorFlow 2 and was trained on a single NVIDIA GeForce GTX TITAN X GPU with a batch size of 16, with a patch size of 256×256 and using the MSE loss function. The parameters of the model were optimized for $10\sim15$ epochs using Adam [20] algorithm with a cosine decay learning rate of $1e{-}4$ to $7e{-}6$. The entire SplitterNet model consists of 731,059 parameters, and it takes 101 ms to process one image of size 2448×3264 pixels on the above-mentioned GPU. The conversion to *TFLite* is done without quantizing the model.

4. Results

In this section, we perform a detailed evaluation of the presented MIDD dataset, and analyze the performance of different models trained on it. Furthermore, we show the quantitative and qualitative results of the proposed SplitterNet model on real-world RGB to RGB denoising problems. The following questions are answered in this section:

1. How well does the proposed model denoise real-world MIDD images in different scenes and conditions.
2. How does the performance of our network compare to SOTA mobile image denoising models as well as models not optimized for mobile usage.

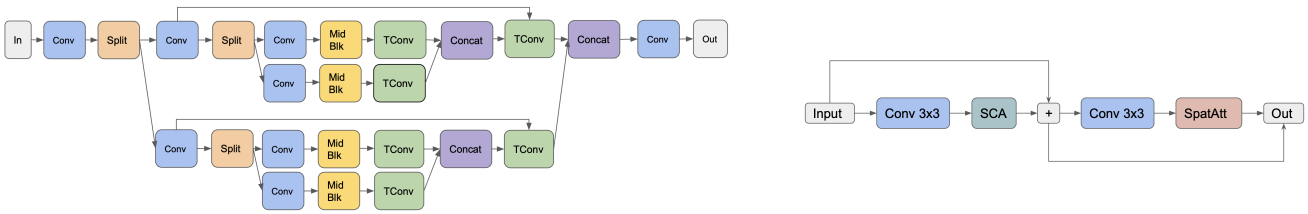

Figure 5. A simplified one-step architecture of the SplitterNet model (left). For each encoding step, the tensor gets split, and in each decoding step, the parts get concatenated again. A representation of the middle block used in the SplitterNet is shown on the right.

3. Does our newly proposed MIDD dataset offer superior data compared to previous datasets, and how do the datasets compare when models are trained on them.
4. How generalizable are the models trained on MIDD.

4.1. Qualitative Evaluation

As one of our goals is to create a mobile denoising model with real-world applicability, we first perform a brief evaluation of its denoising performance on real-world examples from the MIDD dataset. Figure 6 shows the results obtained on several photos taken in different lighting conditions and the corresponding zoomed-in crops for regions of interest. One can see that the model performs well on all conditions and extracts the noise very effectively without introducing any over-smoothing while keeping the sharpness.

4.2. Quantitative Evaluation

In this section, we compare the performance of the SplitterNet model on the MIDD dataset with other architectures developed for this task. We trained various models including NAFNet [12] (in the lightest configuration [1,1,1,1] [1,1,1,1]), U-Net [24], ResNet18 [15], NOAHTCV [16] and Megvii [16]. NOAHTCV and Megvii networks are the winning solutions from the MAI2021 efficient image denoising challenge and are specifically optimized for PSNR, SSIM and runtime efficiency. NAFNet is a recently presented performant baseline for image denoising and reconstruction tasks. The obtained results for these models are shown in Fig. 2, the runtime values were obtained on the Exynos 2200 Mali GPU on images of resolution 720×480 (same setup as in the MAI2021 challenge). One can see that the SplitterNet model achieves higher numerical results compared to the conventional ResNet and U-Net models, while also being 3.5–6 times faster. NAFNet model was too heavy for mobile inference and failed to run on the considered platform. The SplitterNet architecture demonstrated better PSNR and SSIM scores compared to the NOAHTCV network with a similar latency. When compared to the Megvii solution, it achieved a lower PSNR score but a similar SSIM result and a twice lower runtime, thus overall providing a better performance-runtime trade-off, which is a key metric for inference on constrained mobile devices.

Model	PSNR	SSIM	Runtime, *ms*
Identity Mapping	30.88	0.673	-
NAFNet [12]	37.27	0.869	-
ResNet 18 [15]	37.48	0.881	186.0
U-Net [24]	37.87	0.883	95.0
Megvii [16]	**38.05**	**0.884**	56.8
NOAHTCV [16]	37.81	0.883	30.4
SplitterNet (ours)	37.92	**0.884**	**27.7**

Table 2. PSNR, SSIM and runtime results obtained on the MIDD dataset for several image denoising models. The runtime was estimated on the Exynos 2200 Mali GPU on images of size 720×480 px. SplitterNet offers the best performance-runtime trade-off.

4.3. Runtime Evaluation

Next, we check the runtime of the SplitterNet model in real-world conditions when large-resolution input images are used. Same as in the previous section, we test the latency of the model using a publicly available AI Benchmark application[3] [17–19] that can execute custom TensorFlow Lite models on mobile NPUs and GPUs. Table 3 demonstrates the runtime results obtained on all high-end mobile SoCs released in the past 3 years. When executed on mobile GPUs, the latency of the SplitterNet model is less than one second for all mobile SoCs when processing 8MP (4000 × 2000) photos. Moreover, the proposed architecture is also compatible with mobile NPUs available in the last 3 generations of Qualcomm and MediaTek chipsets. The runtime results improve substantially when executed on them: the model achieves a real-time performance (over 30 FPS) for HD-resolution images when deployed on the Snapdragon 8 Gen 3 / 2 and the Dimensity 9000 NPUs, which allows to use it for denoising video data, *e.g.*, in video streaming workloads. Overall, the SplitterNet architecture establishes a very strong baseline for mobile image denoising tasks, not only demonstrating high numerical and visual results but also being able to be deployed on real mobile AI hardware and showing practical latency numbers in this case.

4.4. Cross-Dataset Evaluation

The MIDD dataset was created with the goal of achieving high robustness of trained models when denoising real-

[3] https://ai-benchmark.com/download

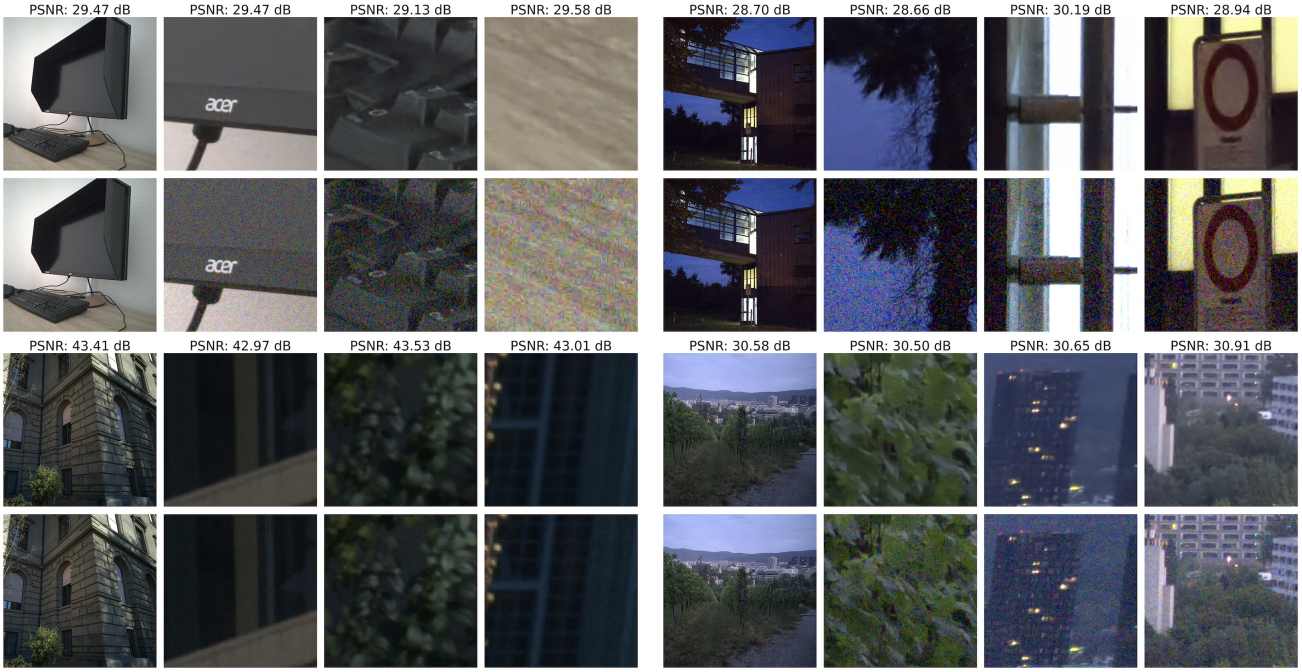

Figure 6. Qualitative comparison of images denoised using the SplitterNet trained on the MIDD. Dark, daylight, cloudy, and indoor scenes get denoised equally well, while the image is not over-smoothed. In the first and third rows, denoised images obtained with our model are presented as well as their PSNR scores. In the second and fourth rows, the original noisy input images are shown.

Mobile Chipset	**Accelerator**	**720×480** ms	**HD** ms	**Full HD** ms	**4MP** ms	**8MP** ms
Qualcomm Snapdragon 8 Gen 3	GPU (Adreno 750)	19	43	91	159	336
Qualcomm Snapdragon 8 Gen 2	GPU (Adreno 740)	21	49	110	195	426
Qualcomm Snapdragon 8 Gen 1	GPU (Adreno 730)	45	100	219	369	869
Qualcomm Snapdragon 888	GPU (Adreno 660)	37	97	220	395	910
MediaTek Dimensity 9300	GPU (Immortalis-G720 MC12)	24	50	103	178	357
MediaTek Dimensity 9200	GPU (Immortalis-G715 MC11)	27	72	130	216	409
MediaTek Dimensity 9000	GPU (Mali-G710 MC10)	32	71	160	271	578
Samsung Exynos 2200	GPU (Xclipse 920)	28	66	144	261	542
Samsung Exynos 2100	GPU (Mali-G78 MP14)	36	86	171	311	642
Google Tensor G2	GPU (Mali-G710 MP7)	75	133	240	374	751
Google Tensor G1	GPU (Mali-G78 MP20)	50	94	172	306	652
Qualcomm Snapdragon 8 Gen 3	NPU (Hexagon HTP Gen 3)	**7.3**	**20**	**48**	110	**257**
Qualcomm Snapdragon 8 Gen 2	NPU (Hexagon HTP Gen 2)	9	24	60	140	310
Qualcomm Snapdragon 8 Gen 1	NPU (Hexagon HTP Gen 1)	20	52	121	249	368
MediaTek Dimensity 9300	NPU (APU 790)	7.7	22	51	**99**	OOM
MediaTek Dimensity 9200	NPU (APU 690)	17	44	100	250	518
MediaTek Dimensity 9000	NPU (APU 590)	22	59	132	191	396

Table 3. Runtime results of the SplitterNet model on various mobile GPUs and NPUs.

world images coming from various mobile sensors. Thus, in this section, we perform several cross-dataset evaluations to check the results in such a setup. Table 4 presents the PSNR scores obtained when training the SplitterNet model on the MAI2021 [16], SIDD [9] and MIDD datasets and evaluating it on the left-out ones. In the rightmost column, all three trained models are evaluated on the previously introduced DPreview dataset containing test image data from 294 different camera sensors. The same training procedures and parameters were used in all cases. When testing on the independent DPreview test set, one can see that the SplitterNet model trained on the MIDD dataset performs best, with a significant advantage of 0.62 dB PSNR over the model trained with the SIDD. The model trained with the MAI2021 dataset performs worst, only achieving a denoising PSNR gap of 4.55. This score indicates that the model trained on the MIDD dataset has the most robust and generalizable image-denoising capabilities.

Trained on	Tested on			
	SIDD	MAI2021	MIDD	DPreview
SIDD [9]	**36.77**	36.00	34.03	21.13
MAI2021 [16]	34.85	37.07	34.69	20.55
MIDD	35.12	**37.23**	**37.92**	**21.75**
Identity	23.66	31.94	30.88	16.00

Table 4. PSNR results of the SplitterNet model in cross-dataset evaluation experiments. The rightmost column shows the evaluation results on the diverse DPreview dataset. The model trained on the MIDD dataset shows the strongest generalization performance as reflected by its score on the independent DPreview dataset.

To further analyze the quality of the dataset, we again performed a cross-dataset experiment, but now with additional fine-tuning on the training parts of the target datasets before getting the results on the corresponding test parts. Table 5 shows that the model trained on the MIDD and fine-tuned on other datasets performs best on each test set. These numbers confirm that the diversity of the MIDD dataset leads to high robustness of the trained models, thus it can be efficiently used for pre-training networks when dealing with the considered and other image restoration tasks.

Trained on	Fine-tuned and Tested on		
	SIDD	MAI2021	MIDD
SIDD	36.77	37.25	37.90
MAI2021	37.98	37.07	37.35
MIDD	**38.79**	**37.62**	**37.92**
Identity	23.66	31.94	30.88

Table 5. PSNR results of the SplitterNet model in cross-dataset evaluation experiments with additional fine-tuning on the training parts of the target datasets. The model trained on the MIDD and fine-tuned on other datasets outperforms the rest of the networks.

We additionally checked the generalization capabilities of the model in the leave-one-out cross-validation. For each run, the model was trained on data from 19 camera sensors and tested on the remaining left-out sensor. The results are presented in Table 6 which is organized as follows: the left column lists the names of camera sensors excluded from the training, the middle column presents the performance scores of the SplitterNet model on this sensor's data, and the right column displays the identity scores thereof. Across each sensor, a significant performance gap is observed when applying the model, indicating that it generalizes effectively and demonstrates strong denoising capabilities on unseen data from different mobile camera sensors.

The visual results of three SplitterNet models trained on the SIDD, MAI2021 and MIDD datasets when denoising DPreview images are analyzed. Fig. 1 shows sample results obtained in this setup. The model trained on the MIDD extracts noise more efficiently while not introducing blur, showing superior denoising performance and confirming the numerical results from the previous experiments.

Sensor	PSNR/SSIM	Id PSNR/SSIM
OV48B	38.73 / 0.912	32.59 / 0.722
OV64B	37.97 / 0.880	30.64 / 0.660
OV32A	39.32 / 0.900	32.57 / 0.730
IMX179	35.26 / 0.792	26.23 / 0.522
IMX258	40.12 / 0.947	33.57 / 0.772
IMX268	40.22 / 0.913	31.66 / 0.674
IMX351	34.92 / 0.863	28.07 / 0.568
IMX362	33.67 / 0.805	25.58 / 0.472
IMX476	39.02 / 0.900	37.15 / 0.867
IMX586	36.32 / 0.900	31.78 / 0.740
IMX686	36.31 / 0.838	28.00 / 0.575
IMX766	39.36 / 0.922	31.93 / 0.687
ISOCELL 3M5	31.63 / 0.792	25.37 / 0.481
ISOCELL GN1	36.41 / 0.817	29.62 / 0.601
ISOCELL 3J1	41.83 / 0.912	36.56 / 0.795
ISOCELL 2L4	37.37 / 0.899	30.41 / 0.683
ISOCELL 3P9	39.66 / 0.949	34.89 / 0.830
S5KHM3	37.98 / 0.912	32.11 / 0.761
S5K3T2	37.52 / 0.854	31.11 / 0.693
Hynix SL846	37.95 / 0.844	29.26 / 0.592

Table 6. Leave-one-out cross-validation results. The SplitterNet was trained on 19 sensors and tested on each left-out sensor. The identity values of the test data are given in the third column.

5. Conclusion

In this paper, we present a new baseline for the mobile image denoising task. As the previously proposed datasets for this problem were either small or contained data only from a few camera sensors, in this work we proposed a novel large-scale MIDD dataset containing over 400K training image pairs collected with 20 different mobile camera sensors in real-life scenarios. The conducted experiments showed that the models trained on this dataset demonstrate much better visual and numerical results when tested on previously unseen images from new sources, thus highlighting its high robustness. Besides that, we presented a novel DPreview test set consisting of noisy / noise-free images from 294 different camera sensors for an extensive benchmarking of image-denoising models. We also proposed an efficient SplitterNet CNN model architecture for the considered task. This model was able to beat the results of the winning solutions from the MAI2021 image denoising challenge in terms of numerical and runtime results. The efficiency of the proposed network was validated on mobile GPUs, where it was able to process 8MP photos under 1 second. Since this model is also compatible with smartphone NPUs, one can further reduce its runtime to 250 ms, or achieve a real-time performance when processing HD-resolution images. We can conclude that the proposed datasets and model establish a strong baseline for the image-denoising task, facilitating further development in this area.

Acknowledgements

This work was partly supported by The Alexander von Humboldt Foundation.

References

[1] Camera2 api. https://developer.android.com/reference/android/hardware/camera2/package-summary. Accessed: 2023-08-06. 3

[2] Dpreview.com. https://www.dpreview.com. Accessed: 2023-08-14. 1, 4, 5

[3] Libraw. https://www.libraw.org. Accessed: 2023-08-31. 4

[4] Mobile statistics report, 2021-2025. https://www.radicati.com/wp/wp-content/uploads/2021/Mobile_Statistics_Report,_2021-2025_Executive_Summary.pdf. Accessed: 2023-08-03. 1

[5] Nnapi. https://developer.android.com/ndk/guides/neuralnetworks. Accessed: 2023-08-04. 2

[6] Tensorflow lite. https://www.tensorflow.org/lite. Accessed: 2023-08-04. 2

[7] Tflite delegates. https://www.tensorflow.org/lite/performance/delegates. Accessed: 2023-08-04. 2

[8] Abdelrahman Abdelhamed, Mahmoud Afifi, Radu Timofte, Michael S. Brown, Yue Cao, Zhilu Zhang, Wangmeng Zuo, Xiaoling Zhang, Jiye Liu, Wendong Chen, Changyuan Wen, Meng Liu, Shuailin Lv, Yunchao Zhang, Zhihong Pan, Baopu Li, Teng Xi, Yanwen Fan, Xiyu Yu, Gang Zhang, Jingtuo Liu, Junyu Han, Errui Ding, Songhyun Yu, Bumjun Park, Jechang Jeong, Shuai Liu, Ziyao Zong, Nan Nan, Chenghua Li, Zengli Yang, Long Bao, Shuangquan Wang, Dongwoon Bai, Jungwon Lee, Youngjung Kim, Kyeongha Rho, Changyeop Shin, Sungho Kim, Pengliang Tang, Yiyun Zhao, Yuqian Zhou, Yuchen Fan, Thomas Huang, Zhihao Li, Nisarg A. Shah, Wei Liu, Qiong Yan, Yuzhi Zhao, Marcin Możejko, Tomasz Latkowski, Lukasz Treszczotko, Michał Szafraniuk, Krzysztof Trojanowski, Yanhong Wu, Pablo Navarrete Michelini, Fengshuo Hu, Yunhua Lu, Sujin Kim, Wonjin Kim, Jaayeon Lee, Jang-Hwan Choi, Magauiya Zhussip, Azamat Khassenov, Jong Hyun Kim, Hwechul Cho, Priya Kansal, Sabari Nathan, Zhangyu Ye, Xiwen Lu, Yaqi Wu, Jiangxin Yang, Yanlong Cao, Siliang Tang, Yanpeng Cao, Matteo Maggioni, Ioannis Marras, Thomas Tanay, Gregory Slabaugh, Youliang Yan, Myungjoo Kang, Han-Soo Choi, Kyungmin Song, Shusong Xu, Xiaomu Lu, Tingniao Wang, Chunxia Lei, Bin Liu, Rajat Gupta, and Vineet Kumar. Ntire 2020 challenge on real image denoising: Dataset, methods and results, 2020. 2

[9] Abdelrahman Abdelhamed, Stephen Lin, and Michael S. Brown. A high-quality denoising dataset for smartphone cameras. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1692–1700, 2018. 1, 2, 4, 7, 8

[10] A. Buades, B. Coll, and J.-M. Morel. A non-local algorithm for image denoising. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 60–65 vol. 2, 2005. 1

[11] A. Buades, B. Coll, and J. M. Morel. A review of image denoising algorithms, with a new one. *Multiscale Modeling & Simulation*, 4(2):490–530, 2005. 1

[12] Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration, 2022. 2, 6

[13] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, 2007. 1

[14] Shuhang Gu and Radu Timofte. A brief review of image denoising algorithms and beyond. In Sergio Escalera, Stephane Ayache, Jun Wan, Meysam Madadi, Umut Güçlü, and Xavier Baró, editors, *Inpainting and Denoising Challenges*, pages 1–21, Cham, 2019. Springer International Publishing. 2

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 6

[16] Andrey Ignatov, Kim Byeoung-su, Radu Timofte, Angeline Pouget, Fenglong Song, Cheng Li, Shuai Xiao, Zhongqian Fu, Matteo Maggioni, Yibin Huang, Shen Cheng, Xin Lu, Yifeng Zhou, Liangyu Chen, Donghao Liu, Xiangyu Zhang, Haoqiang Fan, Jian Sun, Shuaicheng Liu, Minsu Kwon, Myungje Lee, Jaeyoon Yoo, Changbeom Kang, Shinjo Wang, Bin Huang, Tianbao Zhou, Shuai Liu, Lei Lei, Chaoyu Feng, Liguang Huang, Zhikun Lei, and Feifei Chen. Fast camera image denoising on mobile gpus with deep learning, mobile ai 2021 challenge: Report, 2021. 1, 2, 4, 5, 6, 7, 8

[17] Andrey Ignatov, Radu Timofte, William Chou, Ke Wang, Max Wu, Tim Hartley, and Luc Van Gool. Ai benchmark: Running deep neural networks on android smartphones. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018. 6

[18] Andrey Ignatov, Radu Timofte, Andrei Kulik, Seungsoo Yang, Ke Wang, Felix Baum, Max Wu, Lirong Xu, and Luc Van Gool. Ai benchmark: All about deep learning on smartphones in 2019. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3617–3635, 2019. 6

[19] Andrey Ignatov and Radu Timofte et. al. Ai benchmark: An in-depth performance evaluation of all mobile chipsets with ai capabilities in 2024. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2024. 6

[20] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. 5

[21] Yawei Li, Yulun Zhang, Radu Timofte, Luc Van Gool, Zhijun Tu, Kunpeng Du, Hailing Wang, Hanting Chen, Wei Li, Xiaofei Wang, Jie Hu, Yunhe Wang, Xiangyu Kong, Jinlong Wu, Dafeng Zhang, Jianxing Zhang, Shuai Liu, Furui Bai, Chaoyu Feng, Hao Wang, Yuqian Zhang, Guangqi Shao, Xiaotao Wang, Lei Lei, Rongjian Xu, Zhilu Zhang, Yunjin Chen, Dongwei Ren, Wangmeng Zuo, Qi Wu, Mingyan Han, Shen Cheng, Haipeng Li, Ting Jiang, Chengzhi Jiang, Xinpeng Li, Jinting Luo, Wenjie Lin, Lei Yu, Haoqiang Fan, Shuaicheng Liu, Aditya Arora, Syed Waqas Zamir, Javier Vazquez-Corral, Konstantinos G. Derpanis, Michael S. Brown, Hao Li, Zhihao Zhao, Jinshan Pan, Jiangxin Dong, Jinhui Tang, Bo Yang, Jingxiang Chen, Chenghua Li, Xi Zhang, Zhao Zhang, Jiahuan Ren, Zhicheng Ji, Kang Miao, Suiyi Zhao, Huan Zheng, YanYan Wei, Kangliang Liu, Xiangcheng Du, Sijie Liu,

Yingbin Zheng, Xingjiao Wu, Cheng Jin, Rajeev Irny, Sriharsha Koundinya, Vighnesh Kamath, Gaurav Khandelwal, Sunder Ali Khowaja, Jiseok Yoon, Ik Hyun Lee, Shijie Chen, Chengqiang Zhao, Huabin Yang, Zhongjian Zhang, Junjia Huang, and Yanru Zhang. Ntire 2023 challenge on image denoising: Methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 1904–1920, June 2023. 2

[22] Ce Liu, Richard Szeliski, Sing Bing Kang, C. Lawrence Zitnick, and William T. Freeman. Automatic estimation and removal of noise from a single image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):299–314, 2008. 1

[23] Tobias Plötz and Stefan Roth. Benchmarking denoising algorithms with real photographs. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2750–2759, 2017. 2

[24] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015. 6

[25] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration, 2022. 2

[26] Yi Zhang, Dasong Li, Xiaoyu Shi, Dailan He, Kangning Song, Xiaogang Wang, Hongwei Qin, and Hongsheng Li. Kbnet: Kernel basis network for image restoration, 2023. 2