Fast Camera Image Denoising on Mobile GPUs with Deep Learning, Mobile AI 2021 Challenge: Report

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 Feifei Chen

Abstract

Image denoising is one of the most critical problems in mobile photo processing. While many solutions have been proposed for this task, they are usually working with synthetic data and are too computationally expensive to run on mobile devices. To address this problem, we introduce the first Mobile AI challenge, where the target is to develop an end-to-end deep learning-based image denoising solution that can demonstrate high efficiency on smartphone GPUs. For this, the participants were provided with a novel large-scale dataset consisting of noisy-clean image pairs captured in the wild. The runtime of all models was evaluated on the Samsung Exynos 2100 chipset with a powerful Mali GPU capable of accelerating floating-point and quantized neural networks. The proposed solutions are fully compatible with any mobile GPU and are capable of processing 480p resolution images under 40-80 ms while achieving high fidelity results. A detailed description of all models developed in the challenge is provided in this paper.

1. Introduction

Despite the recent advances in mobile camera sensors, image denoising still remains one of the most challenging tasks when it comes to processing mobile photo and video data. The hardware constraints do not allow to significantly increase the size of mobile cameras, which together with increased sensor resolutions and smaller pixels leads to high noise levels on images taken in low-light conditions. To deal with this problem, many classical approaches have been proposed in the past [41, 43, 13, 6, 7, 49, 48, 54]. Much better quantitative results were obtained later with CNN-based deep learning approaches [16, 55, 66, 65, 1, 3]. Despite the good fidelity scores, these works were using either artificial training and validation data [55, 66, 65] or a very small set of indoor images [2, 1, 3], thus limiting their application to real noisy camera data. Besides that, the proposed methods were not optimized for computational efficiency, which is essential for this and other tasks related to image processing and enhancement [22, 23, 37] on mobile devices. In this challenge, we take one step further in solving this problem by using a more advanced real data and by putting additional efficiency-related constraints on the developed solutions.

When it comes to the deployment of AI-based solutions on mobile devices, one needs to take care of the particularities of mobile NPUs and DSPs to design an efficient model. An extensive overview of smartphone AI acceleration hardware and its performance is provided in [33, 30]. According to the results reported in these papers, the latest mobile NPUs are already approaching the results of mid-range desktop GPUs released not long ago. However, there are still two major issues that prevent a straightforward deployment of neural networks on mobile devices: a restricted amount of RAM, and a limited and not always efficient support for many common deep learning layers and operators.

第二个 我理解是 library?

These two problems make it impossible to process high resolution data with standard NN models, thus requiring a careful adaptation of each architecture to the restrictions of mobile AI hardware. Such optimizations can include network pruning and compression [12, 26, 42, 45, 50], 16-bit / 8-bit [12, 40, 39, 62] and low-bit [10, 59, 38, 46] quantization, device- or NPU-specific adaptations, platform-aware neural architecture search [19, 56, 61, 60], *etc*.

---

* Andrey Ignatov, Kim Byeoung-su and Radu Timofte are the Mobile AI 2021 challenge organizers *(andrey@vision.ee.ethz.ch, rui.kim@samsung.com, radu.timofte@vision.ee.ethz.ch)*. Angeline Pouget performed data collection. The other authors participated in the challenge. Appendix A contains the authors' team names and affiliations.

Mobile AI 2021 Workshop website:
https://ai-benchmark.com/workshops/mai/2021/

Figure 1. Sample crops from the original and denoised images from the collected dataset. Best zoomed on screen.

While many challenges and works targeted at efficient deep learning models have been proposed recently, the evaluation of the obtained solutions is generally performed on desktop CPUs and GPUs, making the developed solutions not practical due to the above mentioned issues. To address this problem, we introduce the first *Mobile AI Workshop and Challenges*, where all deep learning solutions are developed for and evaluated on real mobile devices. In this competition, the participating teams were provided with a large-scale image denoising dataset obtained with a recent Sony mobile camera sensor capturing photos in the burst mode. The obtained for each scene images were averaged to get a clean photo, and the resulting noisy-clean image pairs were used to train an end-to-end deep learning solution for this task. Within the challenge, the participants were evaluating the runtime and tuning their models on the Samsung Exynos 2100 platform featuring a powerful Mali-G78 MP14 mobile GPU that can accelerate floating-point and quantized neural networks. The final score of each submitted solution was based on the runtime and fidelity results, thus balancing between the image reconstruction quality and efficiency of the proposed model. Finally, all developed solutions are fully compatible with the TensorFlow Lite framework [57], thus can be deployed and accelerated on any mobile platform providing AI acceleration through the Android Neural Networks API (NNAPI) [4] or custom TFLite delegates [14].

This challenge is a part of the *MAI 2021 Workshop and Challenges* consisting of the following competitions:

- Learned Smartphone ISP on Mobile NPUs [21]
- Real Image Denoising on Mobile GPUs
- Quantized Image Super-Resolution on Mobile NPUs [31]
- Real-Time Video Super-Resolution on Mobile GPUs [28]
- Single-Image Depth Estimation on Mobile Devices [24]
- Quantized Camera Scene Detection on Smartphones [25]
- High Dynamic Range Image Processing on Mobile NPUs

The results obtained in the other competitions and the description of the proposed solutions can be found in the corresponding challenge papers.

2. Challenge

To develop an efficient and practical solution for mobile-related tasks, one needs the following major components:

1. A high-quality and large-scale dataset that can be used to train and evaluate the solution on real (not synthetically generated) data;
2. An efficient way to check the runtime and debug the model locally without any constraints;
3. An ability to regularly test the runtime of the designed neural network on the target mobile platform or device.

This challenge addresses all the above issues. Real training data, tools, and runtime evaluation options provided to the challenge participants are described in the next sections.

2.1. Dataset

To handle the considered image denoising problem, a large scale dataset consisting of noisy-clean images was collected. For this, we used a recent Sony mobile camera sensor and captured photos in the burst mode: for each scene, 20 images were obtained and then averaged to get a clean photo. The illumination conditions varied from moderate to completely dark throughout the data collection process, the images were shot indoors and outdoors to get a variety of in-the-wild scenes with different noise patterns. The photos of more than 1000 different scenes were obtained and then checked manually to remove out-of-focus images and the ones where the built-in optical image stabilization caused misalignments. It should be additionally mentioned that we shot RAW photos using Android Camera2API [11] to avoid

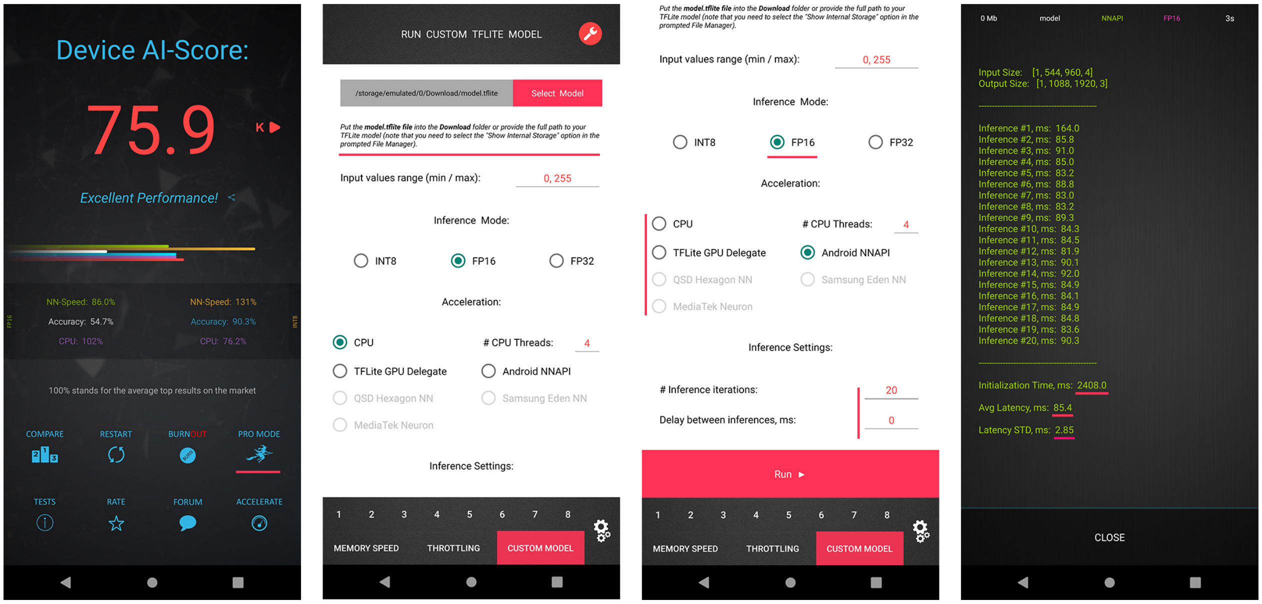

Figure 2. Loading and running custom TensorFlow Lite models with AI Benchmark application. The currently supported acceleration options include Android NNAPI, TFLite GPU, Hexagon NN, Samsung Eden and MediaTek Neuron delegates as well as CPU inference through TFLite or XNNPACK backends. The latest app version can be downloaded at https://ai-benchmark.com/download

any effects from smartphones' integrated image denoising ISP modules, and the resulting images were then converted to RGB format using a stand-alone classical ISP system with disabled noise correction options. The resolution of the images was 4000×3000 pixels, around 650 scenes were used for training the models, while the remaining photos were reserved for validation and testing. An example set of collected images is shown in Fig. 1.

2.2. Local Runtime Evaluation

When developing AI solutions for mobile devices, it is vital to be able to test the designed models and debug all emerging issues locally on available devices. For this, the participants were provided with the *AI Benchmark* application [30, 33] that allows to load any custom TensorFlow Lite model and run it on any Android device with all supported acceleration options. This tool contains the latest versions of *Android NNAPI, TFLite GPU, Hexagon NN, Samsung Eden* and *MediaTek Neuron* delegates, therefore supporting all current mobile platforms and providing the users with the ability to execute neural networks on smartphone NPUs, APUs, DSPs, GPUs and CPUs.

To load and run a custom TensorFlow Lite model, one needs to follow the next steps:

1. Download AI Benchmark from the official website[1] or from the Google Play[2] and run its standard tests.
2. After the end of the tests, enter the *PRO Mode* and select the *Custom Model* tab there.
3. Rename the exported TFLite model to *model.tflite* and put it into the *Download* folder of the device.

只支持到INT8?

4. Select mode type *(INT8, FP16, or FP32)*, the desired acceleration/inference options and run the model.

These steps are also illustrated in Fig. 2.

2.3. Runtime Evaluation on the Target Platform

In this challenge, we use the *Samsung Exynos 2100* SoC as our target runtime evaluation platform. This chipset contains a powerful 14-core *Mali-G78 GPU* capable of accelerating floating point and quantized models, being ranked among the top three mobile platforms by AI Benchmark at the time of its release [5]. Within the challenge, the participants were able to upload their TFLite models to an external server and get a feedback regarding the speed of their model: the runtime of their solution on the above mentioned Mali GPU or an error log if the model contains some incompatible operations. The models were parsed and accelerated using Samsung Eden delegate designed and tuned for high-end Exynos mobile platforms. The same setup was also used for the final runtime evaluation. The participants were additionally provided with a detailed model optimization guideline demonstrating the restrictions and the most efficient setups for each supported TFLite op.

[1] https://ai-benchmark.com/download
[2] https://play.google.com/store/apps/details?id=org.benchmark.demo

Team	Author	Framework	Model Size, KB	PSNR↑	SSIM↑	Runtime, ms ↓	Final Score
NOAHTCV	noahtcv	TensorFlow	209	37.52	0.9150	39	**53.99**
Megvii	chengshen	TensorFlow	14276	37.83	0.9072	84	38.52
ENERZAi Research	Minsu.Kwon	TensorFlow	81	36.33	0.8930	**11**	36.77
MOMA-Denoise	npzl	TensorFlow	1404	37.37	0.9087	54	31.67
ENERZAi Research *	*myungje.lee*	TensorFlow	118	36.22	0.9023	23	15.1
Mier	q935970314	PyTorch / TensorFlow	1528	36.34	0.9066	314	1.31
GdAlgo	TuningMan	PyTorch / TensorFlow	18288	**37.84**	**0.9157**	5019	0.65

Table 1. Mobile AI 2021 Real Image Denoising challenge results and final rankings. The runtime values were obtained on 480p (720×480) images. Team *NOAHTCV* is the challenge winner. * The second solution from *ENERZAi Research* team did not participate in the official test phase, its scores are shown for general information only.

2.4. Challenge Phases

The challenge consisted of the following phases:

I. *Development:* the participants get access to the data and AI Benchmark app, and are able to train the models and evaluate their runtime locally;

II. *Validation:* the participants can upload their models to the remote server to check the fidelity scores on the validation dataset, to get the runtime on the target platform, and to compare their results on the validation leaderboard;

III. *Testing:* the participants submit their final results, codes, TensorFlow Lite models, and factsheets.

2.5. Scoring System

All solutions were evaluated using the following metrics:

- Peak Signal-to-Noise Ratio (PSNR) measuring fidelity score,
- Structural Similarity Index Measure (SSIM), a proxy for perceptual score,
- The runtime on the target Exynos 2100 platform.

The score of each final submission was evaluated based on the next formula (C is a constant normalization factor):

$$\text{Final Score} = \frac{2^{2 \cdot \text{PSNR}}}{C \cdot \text{runtime}},$$

During the final challenge phase, the participants did not have access to the test dataset. Instead, they had to submit their final TensorFlow Lite models that were subsequently used by the challenge organizers to check both the runtime and the fidelity results of each submission under identical conditions. This approach solved all the issues related to model overfitting, reproducibility of the results, and consistency of the obtained runtime/accuracy values.

3. Challenge Results

From above 190 registered participants, 8 teams entered the final phase and submitted valid results, TFLite models, codes, executables and factsheets. Table 1 summarizes the final challenge results and reports PSNR, SSIM and runtime numbers for the top solutions on the final test dataset and on the target evaluation platform. The proposed methods are described in section 4, and the team members and affiliations are listed in Appendix A.

3.1. Results and Discussion

Nearly all submitted solutions demonstrated a very high efficiency: the majority of models are able to process one 480p (720×480 px) image under 60 ms on the target Samsung Exynos 2100 SoC, while the reported runtime results on full-resolution camera images are less than 0.8 seconds for most networks. All proposed solutions were derived from a U-Net [51] like architecture which is not surprising: when feature maps are downsampled in its encoder block, the noise is also removed very efficiently, thus this model type suits perfectly for the considered problem.

用的都基于
U-Net架构哎

The major differences, however, come from the way in which the participants optimized their solutions for better runtime and fidelity results. The challenge winner, team *NOAHTCV*, used Neural Architecture Search (NAS) to find the optimal model design, the same approach was also utilized by *MOMA-Denoise*. *ENERZAi Research* team based its solution on an efficient knowledge transfer approach consisting of the joint training of two (tiny and large) models sharing the same feature extraction block. Their model achieved the fastest runtime and the smallest size (only 81 kilobytes) in this challenge. Another interesting approach was proposed by *Megvii* that presented a modified decoder block splitting and processing feature maps in two parallel channels to reduce the number of multiply-accumulate operations.

The best fidelity scores and visual results were obtained by team *GdAlgo* – this was achieved at the price of using a relatively large multiscale model requiring around five seconds to process one image on Mali GPU. It should be also mentioned that though some of the proposed solutions demonstrated nearly the same speed on desktop CPUs and GPUs, their results on the target Samsung platform differ more than 2-5 times. This explicitly shows that the runtime values obtained on common deep learning hardware are not representative when it comes to model deployment on mobile AI silicon: even solutions that might seem to be very efficient can struggle significantly due to the specific constraints of smartphone AI acceleration hardware and frameworks. This makes deep learning development for mobile devices so challenging, though the results obtained in this competition demonstrate that one can get a very efficient model when taking the above aspects into account.

4. Challenge Methods

This section describes solutions submitted by all teams participating in the final stage of the MAI 2021 Real Image Denoising challenge.

4.1. NOAHTCV

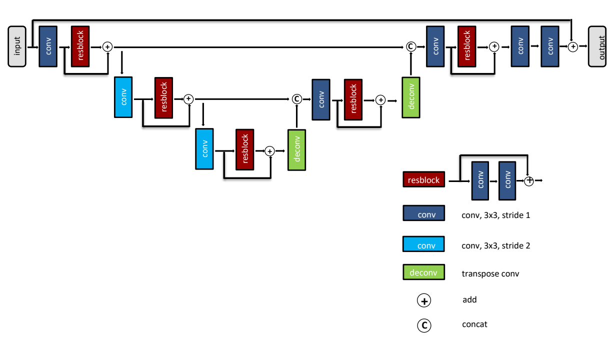

Figure 3. Image denoising network proposed by team NOAHTCV.

Team NOAHTCV applied Neural Architecture Search (NAS) [53] to find an optimal model for this task. The authors started from a multi-scale architecture and used the challenge scoring formula as a target NAS metric. The space of available operators and layers was narrowed to those that are fully supported on mobile devices. To enhance NAS performance, the authors additionally used knowledge distillation to explore more promising candidates and accelerate the optimization procedure. During the fine-tuning stage, each model candidate was optimized by utilizing both the target clean images and the reconstruction results from a larger pre-trained "teacher" model.

Fig. 3 demonstrates the final model architecture. The authors especially emphasize the role of skip connections on fidelity scores and the effect of upsampling and downsampling operations on the runtime results. The models were trained on patches of size 256×256 pixels using Adam optimizer with a batch size of 64 for 200 epochs. The learning rate was set to $1e-4$ and decreased to $1e-5$ by applying a cosine decay. L_2 loss was used as the main fidelity metric.

4.2. Megvii

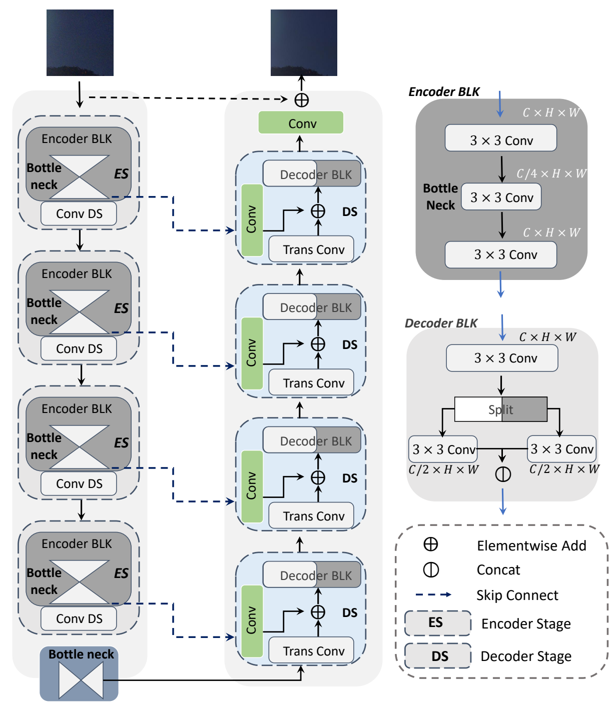

Figure 4. A U-Net based model with modified decoder blocks presented by team Megvii.

Team Megvii proposed a light U-Net [51] based architecture with a modified decoder block (Fig. 4). To improve the efficiency of the model, the authors used small 3×3 convolutional filters in all layers, and set the size of the feature maps to be multiple of eight. No residual blocks were used in the model to decrease its runtime. Each decoder module contains two layers: the output of the first convolutional layer is split into two groups and fed to two convolutions of the second layer, both having half of the channels to decrease the number of FLOPs and multiply-accumulate operations (MACs). The model was trained to maximize PSNR loss using Adam with a batch size of 64 for 162K iterations. The learning rate was set to $2e-4$ and was steadily decreased to $1e-6$ using the cosine annealing strategy. The network was trained on 448×448 patches, vertical and horizontal flips and the Mixup [64] strategy were applied for data augmentation.

4.3. ENERZAi Research

The solution proposed by ENERZAi Research is inspired by the *Once-for-All* approach [8] and consists of two models: one super-network and one sub-network. They both share the same U-Net [51] like module, and the difference comes from their top layers: the sub-network has one

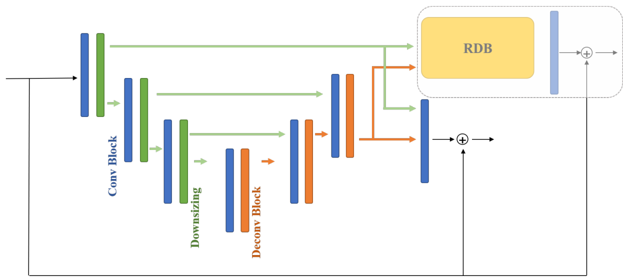

Figure 5. The model architecture proposed by ENERZAi Research team. Semitransparent residual dense block belongs to the super-network and is detached after training.

convolutional layers, while the super-network additionally contains several residual dense blocks as shown in Fig. 5. Both models are first trained jointly using a combination of L_1 and *MS-SSIM* loss functions. The super-network is then detached after the PSNR score goes above a predefined threshold, and the sub-net is further fine-tuned alone. The model was trained on 256×256px patches using Adam optimizer with a batch size of 8 and a learning rate of $1e{-}3$. The resulting model is able to process 2432×3000px images under 300 ms on the Samsung Galaxy S21 smartphone.

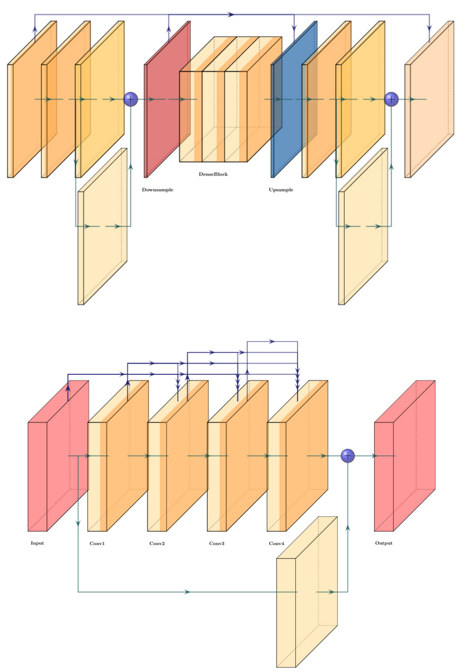

Figure 6. A shallow U-Net based model (top) and a densely connected block (bottom) proposed by ENERZAi Research team.

The second model proposed by this team (which did not officially participate in the final test phase) is demonstrated in Fig. 6. Same as above, the authors started from a standard U-Net based architecture and inserted an additional dense block with skip connections [20] in its bottleneck layer. The authors used *PReLU* activations to get better fidelity results and compressed the model using knowledge distillation technique [18, 17]. The model was trained with a combination of L_1 and *MS-SSIM* losses using a dynamic learning rate [52] ranging from $1e-4$ to $5e-7$.

4.4. MOMA-Denoise

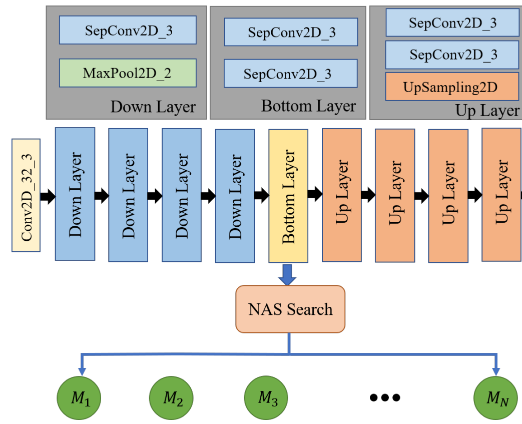

Figure 7. The training NAS-based pipeline and the architecture proposed by MOMA-Denoise team.

MOMA-Denoise team used an in-house Xiaomi AutoML Framework-MOMA to find the best architecture with Neural Architcture Search. The authors used a light-weight U-Net based network consisting of separable convolutions, maxpooling and upsampling layers as a base model, and searched for the best filter and channel sizes using NAS (Fig. 7). To get more training data, the authors unprocessed the provided JPEG images to RAW format, added artificial Poisson-Gaussian noise and mapped the resulting images back to RGB format to mimic real noise present on smartphone photos. The considered image processing pipeline included black level correction, digital gain, demosaicing, device RGB to sRGB mapping, gamma correction and global tone mapping operations. The model was trained to minimize L_1 loss using Adam optimizer with a learning rate of $1e{-}3$ decreased to $1e{-}7$ within the training process.

4.5. Mier

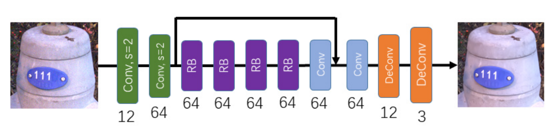

Figure 8. A small U-Net model proposed by Mier team.

Team Mier developed a small U-Net like architecture presented in Fig. 8. This model starts with two convolutions with a stride of 2 to reduce the size of the features, followed by four residual blocks with skip connections and two deconvolutional layers. During the training, the authors used asymmetric convolutions [44, 15] to enhance the kernel skeleton, which were fused after training. The model was trained on 256×256px patches, *Charbonnier* loss was used as a target fidelity metric. The model parameters were optimized using Adam with a batch size of 16 and an initial learning rate of $1e-4$ decreased to $1e-7$ using the cosine annealing strategy. It should be mentioned that the original model was trained in PyTorch and then exported to TensorFlow and TFLite formats.

4.6. GdAlgo

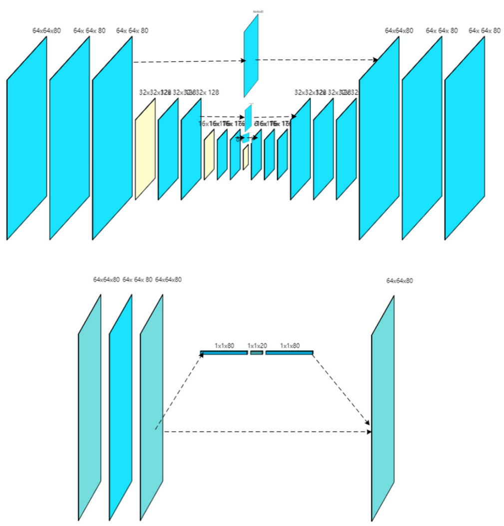

Figure 9. Guided Attention U-Net and a channel attention block designed by GdAlgo team.

Team GdAlgo proposed a Guided Attention U-Net (GAU-net) model demonstrated in Fig. 9. The standard convolutional layers in this model are replaced by channel attention blocks to get better visual and fidelity results. Inspired by the Multi-Stage Image Restoration approach [63], the authors adopted the corresponding multi-stage learning strategy. The model was trained to minimize *Charbonnier* loss function, the edge loss was discarded as no improvement was observed in the experiments.

5. Additional Literature

An overview of the past challenges on mobile-related tasks together with the proposed solutions can be found in the following papers:

- Image Denoising: [1, 3]
- Learned End-to-End ISP: [32, 36]
- Perceptual Image Enhancement: [35, 29]
- Bokeh Effect Rendering: [27, 34]
- Image Super-Resolution: [35, 47, 9, 58]

Acknowledgements

We thank Samsung Electronics, AI Witchlabs and ETH Zurich (Computer Vision Lab), the organizers and sponsors of this Mobile AI 2021 challenge.

A. Teams and Affiliations

Mobile AI 2021 Team

Title:
Mobile AI 2021 Real Image Denoising Challenge
Members:
Andrey Ignatov[1,3] *(andrey@vision.ee.ethz.ch)*, Kim Byeoung-su[2] *(rui.kim@samsung.com)*, Radu Timofte[1,3] *(radu.timofte@vision.ee.ethz.ch)*
Affiliations:
[1] Computer Vision Lab, ETH Zurich, Switzerland
[2] Samsung Electronics, South Korea
[3] AI Witchlabs, Switzerland

NOAHTCV

Title:
Efficient and Specialized Network Search for Image Denoising
Members:
Fenglong Song (songfenglong@huawei.com), Cheng Li, Shuai Xiao, Zhongqian Fu, Matteo Maggioni, Yibin Huang
Affiliations:
Huawei Noah's Ark Lab, China
http://www.noahlab.com.hk/

MegDenoise

Title:
Fast Image Denoise network with Bottleneck Encoder and Slight Decoder
Members:
Shen Cheng (chengshen@megvii.com), Xin Lu, Yifeng Zhou, Liangyu Chen, Donghao Liu, Xiangyu Zhang, Haoqiang Fan, Jian Sun, Shuaicheng Liu
Affiliations:
Megvii, China

ENERZAi Research

Title:
Learning Small Denoising U-Net by Shrinking Large Network
Members:
Minsu Kwon (minsu.kwon@enerzai.com), Myungje Lee, Jaeyoon Yoo, Changbeom Kang, Shinjo Wang

Affiliations:
ENERZAi, Seoul, Korea
enerzai.com

MOMA-Denoise

Title:
UnetS: A Lightweight Denoise Model
Members:
Bin Huang (2659934122@qq.com), Tianbao Zhou
Affiliations:
Xiaomi AI-Lab, China

Mier

Title:
Small UNet
Members:
Shuai Liu[1] *(18601200232@163.com)*, Lei Lei[2], Chaoyu Feng[2]
Affiliations:
[1] North China University of Technology, China
[2] Xiaomi Inc., China

GdAlgo

Title:
Guided Attention U-Net for Image Restoration
Members:
Liguang Huang (huang.liguang@qq.com), Zhikun Lei, Feifei Chen
Affiliations:
Algorithm Department, Goodix Technology, China

References

[1] Abdelrahman Abdelhamed, Mahmoud Afifi, Radu Timofte, and Michael S Brown. Ntire 2020 challenge on real image denoising: Dataset, methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 496–497, 2020. 1, 7

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

[3] Abdelrahman Abdelhamed, Radu Timofte, and Michael S Brown. Ntire 2019 challenge on real image denoising: Methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 1, 7

[4] Android Neural Networks API. `https://developer.android.com/ndk/guides/neuralnetworks`. 2

[5] AI Benchmark Archive. `http://web.archive.org/web/20210425131428/https://ai-benchmark.com/ranking_processors.html`. 3

[6] Antoni Buades, Bartomeu 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. IEEE, 2005. 1

[7] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. A review of image denoising algorithms, with a new one. *Multiscale Modeling & Simulation*, 4(2):490–530, 2005. 1

[8] Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. Once-for-all: Train one network and specialize it for efficient deployment. *arXiv preprint arXiv:1908.09791*, 2019. 5

[9] Jianrui Cai, Shuhang Gu, Radu Timofte, and Lei Zhang. Ntire 2019 challenge on real image super-resolution: Methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 7

[10] Yaohui Cai, Zhewei Yao, Zhen Dong, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Zeroq: A novel zero shot quantization framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13169–13178, 2020. 1

[11] Android Camera2API. `https://developer.android.com/reference/android/hardware/camera2/package-summary`. 2

[12] Cheng-Ming Chiang, Yu Tseng, Yu-Syuan Xu, Hsien-Kai Kuo, Yi-Min Tsai, Guan-Yu Chen, Koan-Sin Tan, Wei-Ting Wang, Yu-Chieh Lin, Shou-Yao Roy Tseng, et al. Deploying image deblurring across mobile devices: A perspective of quality and latency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 502–503, 2020. 1

[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] TensorFlow Lite delegates. `https://www.tensorflow.org/lite/performance/delegates`. 2

[15] Xiaohan Ding, Yuchen Guo, Guiguang Ding, and Jungong Han. Acnet: Strengthening the kernel skeletons for powerful cnn via asymmetric convolution blocks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1911–1920, 2019. 7

[16] Shuhang Gu and Radu Timofte. A brief review of image denoising algorithms and beyond. *Inpainting and Denoising Challenges*, pages 1–21, 2019. 1

[17] Byeongho Heo, Jeesoo Kim, Sangdoo Yun, Hyojin Park, Nojun Kwak, and Jin Young Choi. A comprehensive overhaul of feature distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1921–1930, 2019. 6

[18] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 6

[19] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1314–1324, 2019. 1

[20] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 6

[21] Andrey Ignatov, Jimmy Chiang, Hsien-Kai Kuo, Anastasia Sycheva, and Radu Timofte. Learned smartphone isp on mobile npus with deep learning, mobile ai 2021 challenge: Report. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2021. 2

[22] Andrey Ignatov, Nikolay Kobyshev, Radu Timofte, Kenneth Vanhoey, and Luc Van Gool. Dslr-quality photos on mobile devices with deep convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3277–3285, 2017. 1

[23] Andrey Ignatov, Nikolay Kobyshev, Radu Timofte, Kenneth Vanhoey, and Luc Van Gool. Wespe: weakly supervised photo enhancer for digital cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 691–700, 2018. 1

[24] Andrey Ignatov, Grigory Malivenko, David Plowman, Samarth Shukla, and Radu Timofte. Fast and accurate single-image depth estimation on mobile devices, mobile ai 2021 challenge: Report. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2021. 2

[25] Andrey Ignatov, Grigory Malivenko, and Radu Timofte. Fast and accurate quantized camera scene detection on smartphones, mobile ai 2021 challenge: Report. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2021. 2

[26] Andrey Ignatov, Jagruti Patel, and Radu Timofte. Rendering natural camera bokeh effect with deep learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 418–419, 2020. 1

[27] Andrey Ignatov, Jagruti Patel, Radu Timofte, Bolun Zheng, Xin Ye, Li Huang, Xiang Tian, Saikat Dutta, Kuldeep Purohit, Praveen Kandula, et al. Aim 2019 challenge on bokeh effect synthesis: Methods and results. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3591–3598. IEEE, 2019. 7

[28] Andrey Ignatov, Andres Romero, Heewon Kim, and Radu Timofte. Real-time video super-resolution on smartphones with deep learning, mobile ai 2021 challenge: Report. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2021. 2

[29] Andrey Ignatov and Radu Timofte. Ntire 2019 challenge on image enhancement: Methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 7

[30] 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. 1, 3

[31] Andrey Ignatov, Radu Timofte, Maurizio Denna, and Abdel Younes. Real-time quantized image super-resolution on mobile npus, mobile ai 2021 challenge: Report. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2021. 2

[32] Andrey Ignatov, Radu Timofte, Sung-Jea Ko, Seung-Wook Kim, Kwang-Hyun Uhm, Seo-Won Ji, Sung-Jin Cho, Jun-Pyo Hong, Kangfu Mei, Juncheng Li, et al. Aim 2019 challenge on raw to rgb mapping: Methods and results. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3584–3590. IEEE, 2019. 7

[33] 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. IEEE, 2019. 1, 3

[34] Andrey Ignatov, Radu Timofte, Ming Qian, Congyu Qiao, Jiamin Lin, Zhenyu Guo, Chenghua Li, Cong Leng, Jian Cheng, Juewen Peng, et al. Aim 2020 challenge on rendering realistic bokeh. In *European Conference on Computer Vision*, pages 213–228. Springer, 2020. 7

[35] Andrey Ignatov, Radu Timofte, Thang Van Vu, Tung Minh Luu, Trung X Pham, Cao Van Nguyen, Yongwoo Kim, Jae-Seok Choi, Munchurl Kim, Jie Huang, et al. Pirm challenge on perceptual image enhancement on smartphones: Report. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018. 7

[36] Andrey Ignatov, Radu Timofte, Zhilu Zhang, Ming Liu, Haolin Wang, Wangmeng Zuo, Jiawei Zhang, Ruimao Zhang, Zhanglin Peng, Sijie Ren, et al. Aim 2020 challenge on learned image signal processing pipeline. *arXiv preprint arXiv:2011.04994*, 2020. 7

[37] Andrey Ignatov, Luc Van Gool, and Radu Timofte. Replacing mobile camera isp with a single deep learning model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 536–537, 2020. 1

[38] Dmitry Ignatov and Andrey Ignatov. Controlling information capacity of binary neural network. *Pattern Recognition Letters*, 138:276–281, 2020. 1

[39] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2704–2713, 2018. 1

[40] Sambhav R Jain, Albert Gural, Michael Wu, and Chris H Dick. Trained quantization thresholds for accurate and efficient fixed-point inference of deep neural networks. *arXiv preprint arXiv:1903.08066*, 2019. 1

[41] Darwin T Kuan, Alexander A Sawchuk, Timothy C Strand, and Pierre Chavel. Adaptive noise smoothing filter for images with signal-dependent noise. *IEEE transactions on pat-*

tern analysis and machine intelligence, (2):165–177, 1985. 1

[42] Yawei Li, Shuhang Gu, Luc Van Gool, and Radu Timofte. Learning filter basis for convolutional neural network compression. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5623–5632, 2019. 1

[43] 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, 2007. 1

[44] Shuai Liu, Chenghua Li, Nan Nan, Ziyao Zong, and Ruixia Song. Mmdm: Multi-frame and multi-scale for image demoiréing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 434–435, 2020. 7

[45] Zechun Liu, Haoyuan Mu, Xiangyu Zhang, Zichao Guo, Xin Yang, Kwang-Ting Cheng, and Jian Sun. Metapruning: Meta learning for automatic neural network channel pruning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3296–3305, 2019. 1

[46] Zechun Liu, Baoyuan Wu, Wenhan Luo, Xin Yang, Wei Liu, and Kwang-Ting Cheng. Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. In *Proceedings of the European conference on computer vision (ECCV)*, pages 722–737, 2018. 1

[47] Andreas Lugmayr, Martin Danelljan, and Radu Timofte. Ntire 2020 challenge on real-world image super-resolution: Methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 494–495, 2020. 7

[48] M Kivanc Mihcak, Igor Kozintsev, Kannan Ramchandran, and Pierre Moulin. Low-complexity image denoising based on statistical modeling of wavelet coefficients. *IEEE Signal Processing Letters*, 6(12):300–303, 1999. 1

[49] Mukesh C Motwani, Mukesh C Gadiya, Rakhi C Motwani, and Frederick C Harris. Survey of image denoising techniques. In *Proceedings of GSPX*, volume 27, pages 27–30, 2004. 1

[50] Anton Obukhov, Maxim Rakhuba, Stamatios Georgoulis, Menelaos Kanakis, Dengxin Dai, and Luc Van Gool. T-basis: a compact representation for neural networks. In *International Conference on Machine Learning*, pages 7392–7404. PMLR, 2020. 1

[51] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 4, 5

[52] Leslie N Smith. Cyclical learning rates for training neural networks. In *2017 IEEE winter conference on applications of computer vision (WACV)*, pages 464–472. IEEE, 2017. 6

[53] Dehua Song, Chang Xu, Xu Jia, Yiyi Chen, Chunjing Xu, and Yunhe Wang. Efficient residual dense block search for image super-resolution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12007–12014, 2020. 5

[54] Jean-Luc Starck, Emmanuel J Candès, and David L Donoho. The curvelet transform for image denoising. *IEEE Transactions on image processing*, 11(6):670–684, 2002. 1

[55] Ying Tai, Jian Yang, Xiaoming Liu, and Chunyan Xu. Memnet: A persistent memory network for image restoration. In *Proceedings of the IEEE international conference on computer vision*, pages 4539–4547, 2017. 1

[56] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2820–2828, 2019. 1

[57] TensorFlow-Lite. `https://www.tensorflow.org/lite`. 2

[58] Radu Timofte, Shuhang Gu, Jiqing Wu, and Luc Van Gool. Ntire 2018 challenge on single image super-resolution: Methods and results. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 852–863, 2018. 7

[59] Stefan Uhlich, Lukas Mauch, Fabien Cardinaux, Kazuki Yoshiyama, Javier Alonso Garcia, Stephen Tiedemann, Thomas Kemp, and Akira Nakamura. Mixed precision dnns: All you need is a good parametrization. *arXiv preprint arXiv:1905.11452*, 2019. 1

[60] Alvin Wan, Xiaoliang Dai, Peizhao Zhang, Zijian He, Yuandong Tian, Saining Xie, Bichen Wu, Matthew Yu, Tao Xu, Kan Chen, et al. Fbnetv2: Differentiable neural architecture search for spatial and channel dimensions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12965–12974, 2020. 1

[61] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10734–10742, 2019. 1

[62] Jiwei Yang, Xu Shen, Jun Xing, Xinmei Tian, Houqiang Li, Bing Deng, Jianqiang Huang, and Xian-sheng Hua. Quantization networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7308–7316, 2019. 1

[63] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Multi-stage progressive image restoration. *arXiv preprint arXiv:2102.02808*, 2021. 7

[64] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 5

[65] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing*, 26(7):3142–3155, 2017. 1

[66] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Ffdnet: Toward a fast and flexible solution for cnn-based image denoising. *IEEE Transactions on Image Processing*, 27(9):4608–4622, 2018. 1