This CVPR Workshop paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Learned Smartphone ISP on Mobile GPUs, Mobile AI 2025 Challenge: Report

Andrey Ignatov Georgy Perevozchikov Radu Timofte Cheng Li Lian Liu
Jun Cao Heng Sun Wu Pan Song Wang KeQiang Yu Shuo Liu
HongQin He ZhenHao Dong JianKe Chen DeJun Hao Keqiang Yu
Tingniao Wang Xiaoqing Zhou Dong Zhang Chunxia Lei Jianke Chen
Dejun Hao Jianguang He Hailong Yan Ao Li Xiangtao Zhang Zhe Liu
Ce Zhu Le Zhang Andrei Arhire Shuo Liu Junpyo Seo Zhenhao Dong
Jianke Chen Fen Xie Xiuzhi Fang Chen Wu Zhansheng Wang Pengbo Zhang
Jiazi Huang

Abstract

RGB photo reconstruction from RAW camera images is an increasingly popular deep learning problem with a practical application to mobile cameras. This creates a need for solutions that are not only performant but are additionally compatible with real mobile AI hardware such as GPUs or NPUs. In this Mobile AI challenge, we address this problem and propose the participants to design efficient learned ISP models that can demonstrate fast inference times on mobile GPUs. For this, the participants were provided with a large-scale Fujifilm UltraISP dataset consisting of RAW-RGB image pairs captured with the Sony IMX586 Quad Bayer mobile sensor and a professional 102MP medium format FujiFilm GFX100 camera. The runtime of all models was evaluated on the latest Adreno and Mali GPUs used in Qualcomm and MediaTek chipsets. The proposed solutions are compatible with all recent mobile GPUs, being able to process Full HD photos under 30 ms in the majority of cases and delivering high-fidelity results. A comprehensive description of the models developed in the challenge is provided in this paper.

1. Introduction

As smartphone cameras continue to improve, much of the progress is driven by advanced image processing algorithms, such as those used for noise reduction, accurate color reproduction and high dynamic range (HDR) processing. While one can potentially replace each classical image signal processing (ISP) step with an advanced deep learning model to get a better reconstruction result, a more radical approach is to replace the entire RAW-to-RGB ISP pipeline with a single neural network that will inherently learn all the underlying image manipulation stages. Such approach will also provide the model with a reacher 10–16 bit information stored in the original unaltered RAW bayer data, which can further help to improve the image restoration quality. In [34], the authors first proposed this approach: they collected Zurich RAW-to-RGB dataset, which consists of RAW-RGB image pairs captured by both a mobile camera sensor and a high-end DSLR, and then trained a deep neural network on this data. The results demonstrated by the authors showed that this approach could potentially outperform the commercial hand-crafted ISP of the Huawei P20 smartphone. Even better results were achieved by the models proposed in the follow-up challenges using the same dataset [14, 29, 33, 52, 71]. The next step in building high-quality deep learning-based ISPs was done in [42, 43], where the authors presented a more advanced Fujifilm UltraISP dataset for the considered task and mobile-friendly CNN architectures that were able to process high-resolution images directly on smartphones. In this competition, we use the same dataset and propose the participants to design even more efficient learned ISP models that can run on recent smartphone GPUs.

The development of mobile-friendly deep learning architectures that can be executed on various AI accelerators including NPUs is not a straightforward task. It requires a deep understanding of numerous limitations imposed by mobile AI hardware such as limited support for ML oper-

*Andrey Ignatov, Georgy Perevozchikov and Radu Timofte are the organizers of the Mobile AI 2025 challenge *(andrey@vision.ee.ethz.ch, georgii.perevozchikov@uni-wuerzburg.de, radu.timofte@uni-wuerzburg.de)*. The other authors participated in the challenge.
Appendix A contains the authors' team names and affiliations.

Mobile AI 2025 Workshop website:
https://ai-benchmark.com/workshops/mai/2025/



Figure 1. A sample set of images from the collected dataset. From left to right: the original RAW image visualized with Photoshop's built-in raw image processing engine, RGB image obtained with MediaTek's built-in ISP system, and the target Fujifilm photo.

ators, restricted memory size and bandwidth, accuracy issues for quantized models, specific NPU driver properties, *etc*. Even a supported ML layer or operator might not be efficiently executed on NPUs, becoming a major bottleneck during model inference, and thus requiring partial or even full architecture re-design in order to achieve lower latency and energy consumption. We refer to [26, 30] for a more detailed overview of the particularities and performance evaluation of recent mobile AI accelerators. The above mentioned limitations make the inference with standard deep learning architectures either inefficient or not even possible at all on mobile NPUs, therefore requiring various adaptations and low-level model design optimizations including 16-bit / 8-bit [5, 11, 50, 51, 59, 60, 65, 70, 72, 84] and low-bit [4, 6, 9, 49, 61, 76, 89] quantization, network pruning and compression [11, 19, 53, 54, 58, 62, 66, 79, 83, 88], NPU-specific adaptations, platform-aware neural architecture search [12, 13, 21, 64, 73, 77, 81, 82, 85, 86], *etc*.

In this Mobile AI Challenge, the participants were proposed to develop efficient learned ISP models that are compatible with smartphone GPUs. The Fujifilm UltraISP dataset [43] consisting of thousands paired photos captured with a normal mobile camera sensor and a professional 102MP medium-format FujiFilm GFX100 camera was considered in this competition. The runtime of the models was evaluated on Adreno and Mali mobile GPUs found in the latest Qualcomm and MediaTek SoCs. Each submission was evaluated based on both image fidelity and runtime efficiency, thus balancing reconstruction quality with model efficiency. All solutions developed during this challenge are fully compatible with the LiteRT [67] (rebranded TensorFlow Lite) framework, enabling deployment on a variety of Linux and Android-based IoT platforms, smartphones, and edge devices.

This challenge is a part of the *MAI 2025 Workshop and Challenges* consisting of the following competitions:

- Quantized Image Super-Resolution on Mobile NPUs [48]
- Real-Time Video Super-Resolution on Mobile NPUs
- RGB Photo Enhancement on Mobile GPUs [47]
- Learned Smartphone ISP on Mobile GPUs
- Bokeh Effect Rendering on Mobile GPUs
- Image Denoising on Mobile GPUs
- Real-Time Camera Scene Detection on Mobile NPUs

The results and solutions obtained in the previous *Mobile AI Challenges* are described in the following papers:

- Quantized Image Super-Resolution on Mobile NPUs [40, 41]
- Learned Smartphone ISP on Mobile NPUs / GPUs [36, 45]
- Video Super-Resolution on Mobile GPUs [39, 44]
- Image Denoising on Mobile GPUs [35]
- Single-Image Depth Estimation on Mobile Devices [37, 41]
- Bokeh Effect Rendering on Mobile GPUs [46]
- Camera Scene Detection on Smartphones [38, 68]

2. Challenge

To design an efficient and practical solution for mobile-related tasks, the following key components are essential:

1. A high-quality and large-scale dataset that can be used to train and evaluate the solution;
2. An efficient way to check the runtime and debug the model locally without any constraints.

This challenge addresses 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

In this challenge, we use the Fujifilm UltraISP dataset [42, 43] collected using the Fujifilm GFX100 medium format 102 MP camera capturing the target high-quality images, and a popular Sony IMX586 Quad Bayer mobile camera sensor that can be found in tens of mid-range and high-end mobile devices released during the past years. The Sony sensor was mounted on the MediaTek Dimensity 820 development board, and was capturing both raw and processed (by its built-in ISP system) 12MP images. The Dimensity board was rigidly attached to the Fujifilm camera, and they were shooting photos synchronously to ensure that the image

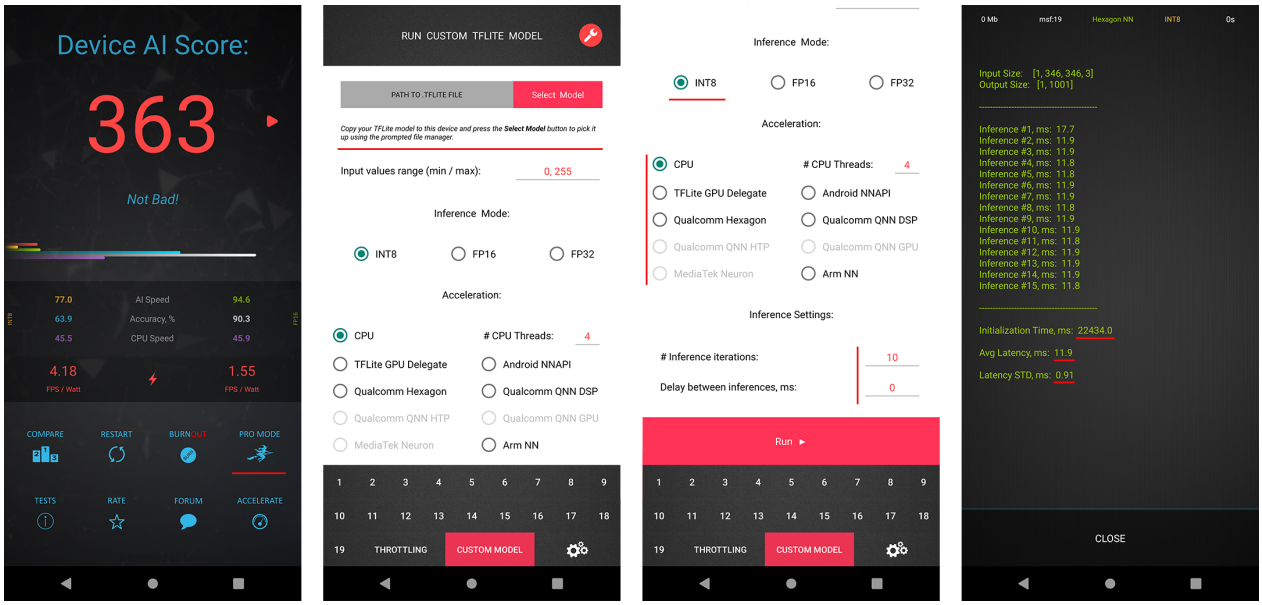


Figure 2. Loading and running custom TensorFlow Lite models with AI Benchmark application. The currently supported acceleration options include Android NNAPI, TFLite GPU, Hexagon NN, Samsung Eden and MediaTek Neuron delegates as well as CPU inference through TFLite or XNNPACK backends. The latest app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

content is identical. The dataset contains over 6 thousand daytime image pairs captured at a wide variety of places with different illumination and weather conditions. An example set of full-resolution photos from the Fujifilm UltraISP dataset is shown in Fig. 1. As the collected RAW-RGB image pairs were not perfectly aligned, they were initially matched using the state-of-the-art deep learning based dense matching algorithm [75] to extract 256×256 pixel patches from the original photos. It should be mentioned that all alignment operations were performed on Fujifilm RGB images only, therefore RAW photos from the Sony sensor remained unmodified, exhibiting the same values as read from the sensor.

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 [26, 30] 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 *Qualcomm QNN, Hexagon NN, MediaTek Neuron, TFLite GPU, Android NNAPI* and *ARM NN* 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 (LiteRT) 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. Put the exported TFLite (LiteRT) model into the *Download* folder of the device.
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. The instructions on how to generate TFLite model from PyTorch, TensorFlow and Keras pre-trained models were provided in the MAI Workshop GitHub repository [3].

2.3. Runtime Evaluation on the Target Platform

In this challenge, we use the *Adreno 830* and *Arm Immortalis-G925* (12-core) mobile GPUs that can be found in the *Qualcomm Snapdragon 8 Elite* and *MediaTek Dimensity 9400* mobile SoCs as our target runtime evaluation platforms. The considered GPUs can efficiently accelerate floating-point networks and demonstrate the runtimes that are comparable to the ones obtained on mid-range NPUs. The models were parsed and accelerated using the TFLite GPU delegate [56] showing the best performance on these GPUs. All final solutions were tested using the aforementioned AI Benchmark application.

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

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

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

[3] https://github.com/aiff22/MAI-2025-Workshop

Team	Author	Framework	Model Size, KB	PSNR↑	SSIM↑	Adreno GPU, ms ↓	Arm GPU, ms ↓	Final Score
AwesomeNeural	lc	PyTorch	18 Kb	24.09	0.8566	4.9	24.8	**16.5**
DHTXSF-Lab	DahuaISP	PyTorch	20 Kb	24	0.857	8.6	27.4	10.4
SHBD	LucasH	PyTorch	25 Kb	23.9	0.8515	11.6	30.8	7.4
AVC2	AVC2	PyTorch	23 Kb	**24.22**	0.8581	22.2	46.7	6.7
UAIC ISP, Track 1	AndreiArhire	Keras / TensorFlow	6 Kb	23.01	0.8306	**3.26**	**16.6**	5.5
DHTSDX	dhtsdx	PyTorch	36 Kb	23.76	0.8493	20.7	46	3.7
SJP-ISP	pyojuncode	Keras / TensorFlow	163 Kb	23.14	0.8344	15.4	31.8	2.2
UAIC ISP, Track 2	AndreiArhire	Keras / TensorFlow	56 Mb	23.84	0.8602	90.3	185	1
D-Image	Dzhh	PyTorch / Keras	1.2 Mb	24.04	**0.8671**	1526	1821	0.1
oteam	wangty	PyTorch	211 Mb	23.52	0.8603	6787	6590	0
MicroISP [43]	-	Keras / TensorFlow	155 Kb	23.87	0.853	23	29	5.2
PyNET-V2 Mobile [42]	-	Keras / TensorFlow	3.6 Mb	24.72	0.8783	81.3	142	4
PyNET [34]	-	Keras / TensorFlow	181 Mb	25.01	0.8985	3378	4736	0.2

Table 1. Mobile AI 2025 smartphone ISP challenge results and final rankings. The runtime values were obtained on Full HD (1920×1088) images. Team *AwesomeNeural* is the challenge winner. The results of the MicroISP and PyNET(-V2) models are provided for the reference.

dataset, to get the runtime on the target platform, and to compare their results on the validation leaderboard;

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

2.5. Scoring System

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, and to compare their results on the validation leaderboard;

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

2.6. 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 mobile GPUs.

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 70 registered participants, 9 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 each submitted solution 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

All submitted solutions demonstrated a very high efficiency: the majority of models are able to process one Full HD (1920×1088 px) image under 50 ms on the target GPUs. This is a major improvement compared to the results obtained in the MAI 2021 Learned ISP challenge [36], where the best runtime achieved on the Dimensity 1000+ NPU was 61ms. This shows great progress in both model design as well as in mobile AI acceleration hardware that is currently able to provide decent speed-up for the most common network layers. Team *AwesomeNeural* is the challenge winner: solution proposed by this team was able to demonstrate over 200 and 40 FPS when running on the Adreno and Arm GPUs, respectively. This highlights that the considered model can be potentially applied even for Full HD RAW video processing while achieving a frame rate of over 30 FPS on all current mid-range and high-end smartphone SoCs. Very similar numerical, runtime and visual results were achieved by team *DHTXSF-Lab*, which solution was slightly slower on both platforms. The most efficient model design was proposed team *UAIC ISP*: with a model size of *6Kb*, this solution demonstrated 300 and 60 FPS on the Adreno and Mali GPUs, respectively. Such tiny CNN architecture consisting of only three convolutional layers should be deployable even on low-power IoT NPUs while still providing decent image reconstruction visual results.

A notable progress that should be highlighted is the reduced complexity of model conversion to the target TFLite format. Table 1 shows that the majority of solutions in this challenge were developed using the PyTorch ML framework. Due to the recently presented *ai-edge-torch*[4] plugin developed by the TFLite (LiteRT) team, one can now directly convert the majority of PyTorch models to the TFLite format without a need for TensorFlow / Keras model re-implementation or use of ONNX as an intermediary format. This significantly simplifies the development process of mobile-friendly deep learning architectures for the majority of researchers

[4] https://github.com/google-ai-edge/ai-edge-torch

who use PyTorch as their main framework for AI model development and training.

The last thing that should be mentioned is that there still exists a large gap in visual results between all solutions proposed in this challenge and more performant models [34, 42]: in terms of texture, color, noise and dynamic range image quality aspects, the latter ones demonstrate substantially better results, which is also reflected by SSIM and PSNR scores. We hope that the next challenges and works on the RAW-to-RGB mapping task will bridge this gap, presenting both efficient and robust mobile-friendly learned ISP architectures.

4. Challenge Methods

This section describes solutions submitted by all teams participating in the final stage of the MAI 2025 Learned Smartphone ISP challenge.

4.1. AwesomeNeural

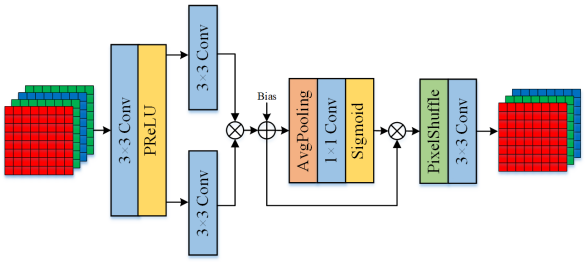


Figure 3. Nano Vision Net (NVNet) architecture proposed by team AwesomeNeural

Team AwesomeNeural introduced the Nano Vision Net (NVNet), presented in Fig. 3, which is an optimized architecture based on SYENet [18]. The number of channels in the team's network is 12. The network commences with a 3×3 convolutional layer followed by a PReLU activation for initial feature extraction, replacing a more complex head structure from the original SYENet. The subsequent features are processed through two parallel 3×3 convolutional layers. The outputs of these parallel paths undergo element-wise multiplication, after which a bias term is added. A channel attention mechanism is then applied; specifically, an attention map derived from the initial features using Average Pooling, a 1×1 convolution, and a Sigmoid function is element-wise multiplied with the processed features from the parallel paths. Finally, a PixelShuffle layer performs $2\times$ upsampling, followed by a 3×3 convolution to output the RGB image. All convolutional layers in the network employ the reparameterization technique. The structural modifications were implemented to decrease computation time relative to SYENet while maintaining comparable performance.

The team adopted the same training strategy and reparameterization technique as specified for SYENet [18]. The entire dataset was used for training with a batch size of 16.

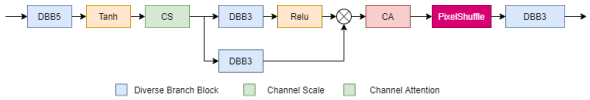


Figure 4. DBF-Net architecture proposed by DHTXSF-Lab team

4.2. DHTXSF-Lab

Inspired by structural reparameterization techniques and previous work [15, 45], the DHTXSF-Lab team proposed the DBF-Net architecture, illustrated in figure 4. The network utilizes an expanded version of the Diverse Branch Block (DBB) [15] as its core component, aiming to enhance multiscale feature capture during training, using a wide variety of convolution layers with different kernels (Fig. 5) and allow for efficient block fusion during inference. The network starts with a DBB block, followed by a Tanh activation and a Channel Scale (CS) layer [22], which applies learnable scale and bias factors per channel. The main path proceeds through another DBB with a ReLU activation. In parallel, a separate branch uses a DBB originating from the CS output to estimate spatial attention weights. These weights are then element-wise multiplied with the main path features. Following this, a Channel Attention (CA) module is applied for adaptive color equalization. Upsampling is performed using a PixelShuffle layer, and a final DBB block conducts fine-tuning before the output.

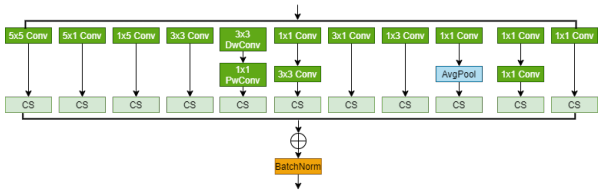


Figure 5. Diverse Branch Block (DBB) architecture

The training strategy involved two loss phases: an initial pre-training phase using L_1 loss, followed by a fine-tuning phase where a perceptual loss was added to the L_1 loss. Model parameters were optimized using the Adam algorithm with a batch size of 32. The learning rate was initialized at 1e-4 and was decreased throughout training.

4.3. SHBD

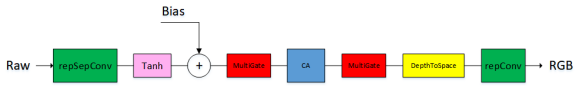


Figure 6. The overall model architecture proposed by the SHBD team.

Team SHBD introduced a network (Fig. 6) incorporating their novel MultiGate nonlinear activation module, inspired by GELU and Simplified Gate [10, 20, 45], to enhance network expressiveness. As shown in Figure 7, the MultiGate module processes an

input via three parallel branches – a 3×3 convolution, a 3×3 depthwise convolution, and a 1×1 convolution – each followed by a Sigmoid activation. The outputs of these three activated paths are element-wise multiplied together, and this result is then element-wise multiplied by the original input. The overall network architecture (Fig. 6) starts with a reparameterizable separable convolution (repSepConv) and a Tanh activation, followed by a bias addition. This output passes through the first MultiGate module, then a Channel Attention (CA) block, and subsequently a second MultiGate module. Upsampling is performed via a DepthToSpace layer, followed by a final reparameterizable convolution (repConv) to yield the RGB result. Both repSepConv and repConv blocks are designed to fuse into standard 3×3 convolutions for efficient inference.

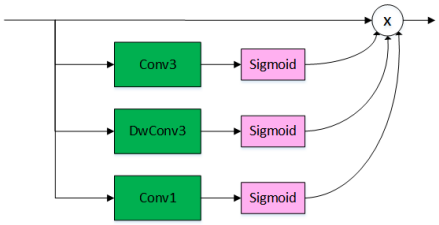


Figure 7. MultiGate block.

The model was trained using a composite loss function combining L_2 loss, SSIM loss, and a color average loss. The training strategy emphasized learning from samples with high initial losses and employed different loss weights for flat versus edge regions to handle potential data misalignment. Optimization was performed using the Adam optimizer with a batch size of 96 and an initial learning rate of 1e-4, which decreased during the training process.

4.4. AVC2

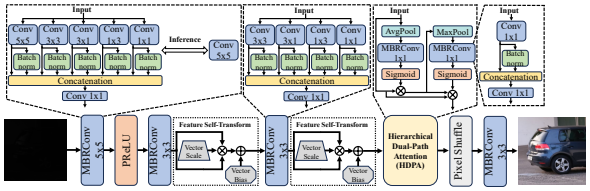


Figure 8. General method proposed by AVC2 team

Team AVC2 introduced MobileIE, a lightweight image signal processing network designed for mobile deployment that balances performance and efficiency using reparameterization, as illustrated in Fig. 8. The architecture begins with an MBRConv 5×5 layer followed by a PReLU activation. The subsequent main body consists of two sequential blocks; each block contains an MBRConv 3×3 layer followed by a Feature Self-Transform (FST) module, incorporating residual connections and a bias vector. After these blocks, a Hierarchical Dual-Path Attention (HDPA) module is employed to refine features. Upsampling is achieved through a Pixel Shuffle layer, followed by a final MBRConv 3×3 layer generating the output image. A key aspect is the MBRConv block, which utilizes multiple parallel convolutional branches with batch normalization during training; these blocks are reparameterized into standard single convolutional layers for efficient inference.

The network was trained using the L_1 loss function, favored for promoting sharper image reconstruction. The Adam optimizer was utilized with a cosine annealing learning rate schedule. The schedule started with an initial learning rate of 0.001, preceded by a 10-epoch warm-up phase at a fixed rate of 1e-6. The learning rate was subsequently reset every 50 epochs before decaying. The model was trained for a total of 5,000 epochs with a batch size of 32. The training employed 24,000 aligned raw-sRGB image pairs, partitioned into 22,000 for training and 2,000 for testing.

4.5. UAIC_ISP

The UAIC_ISP team presented distinct solutions [3] for the challenge's two tracks. Below is a detailed description of each track.

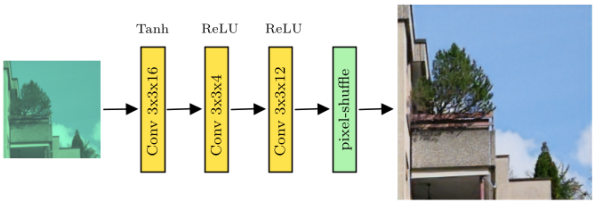


Figure 9. The CNN architecture proposed by the UAIC_ISP team.

Track 1: For Track 1, the team developed a compact model (Fig. 9) inspired by the architectures of dh_isp [36] and MiAlgo [45], aiming to match the fidelity of the latter with reduced parameters. The network comprises three sequential 3×3 convolutional layers, with channel counts of 16, 4, and 12, respectively. Tanh activation follows the first layer, while ReLU is used after the second and third layers. Upsampling is performed by a final PixelShuffle layer. The channel counts were chosen considering the minimum requirement for PixelShuffle (12) and experimental findings that the middle layer was less sensitive to reduction. To address potential data misalignment, training utilized paired data where raw images (specifically from the Fujifilm UltraISP dataset) were matched with outputs from a separately pre-trained large RMFA-Net [57] (20 blocks, width 16).

The proposed compact model was trained for 300 epochs using L_1 loss and the Adam optimizer with a batch size of 32. A cosine learning rate scheduler adjusted the rate between 10^{-3} and 10^{-6}. Data augmentation involved random rotations and flips applied as subsets of data were loaded into memory. Custom validation and test sets of 1024 samples each were created for experimentation.

Track 2: The solution for Track 2 (Fig. 10) employed the configuration of RMFA-Net [57] with 2 blocks and a channel width of 16. The objective was to produce results with high perceptual quality and efficiently executable on smartphone GPUs. The RMFA-Net processes an input image using an input module, followed by multiple stacked RMFA blocks connected with residual additions, and concludes with an output module. Each RMFA block internally performs complex feature refinement using multi-

scale processing (via pooling and upsampling), skip connections, and both channel and spatial attention mechanisms.

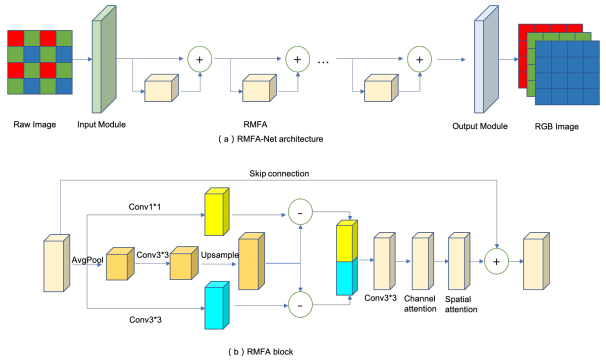


Figure 10. The RMFA-Net architecture [57] used by the UAIC_ISP team.

For training, 17.6% of the provided training pairs were manually excluded. A combination of L_1, SSIM, and LPIPS [87] losses guided the training. An adaptive loss weighting strategy was implemented where each loss component's coefficient was adjusted per step to maintain a gradient norm of 1 during updates. The model was trained using the Adam optimizer with a batch size of 8 and a cosine learning rate scheduler.

4.6. DHTSDX

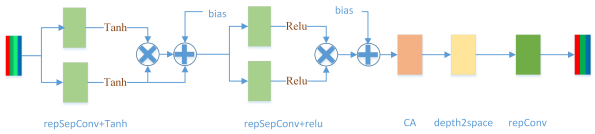


Figure 11. The CNN architecture proposed by the DHTSDX team.

The architecture presented by team DHTSDX (Fig. 11) adapts the structure from the MAI2022 Multimedia team [45] by incorporating custom reparameterizable separable convolution blocks (repSepConv, Fig. 12). The network begins with two parallel repSepConv blocks, each followed by Tanh activation; their outputs are element-wise multiplied and combined with a bias term. This structure is repeated, following this, a Channel Attention (CA) module refines the features, which are then upsampled using DepthToSpace. A final reparameterizable standard convolution (repConv) produces the output. The repSepConv block (training phase) features multiple parallel depthwise convolutional branches (varying kernel sizes), each followed by a Channel Scale module [22]. The summed output of these branches passes through a 1×1 convolution and a BatchNorm layer. For inference, each repSepConv block fuses into efficient depthwise separable convolutions.

The training process consisted of two stages. The pre-training employed the standard L_1 loss. The subsequent fine-tuning stage utilized data selection, using only training samples with PSNR below a dynamically updated threshold. This stage used a custom

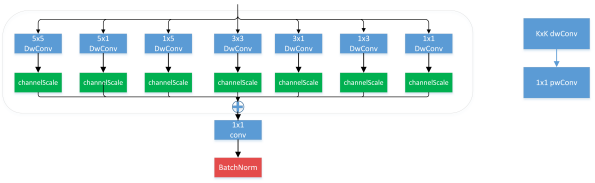


Figure 12. The structure of the repSepConv block during the training (left) and inference (right) stages.

PSNR-weighted loss function:

$$L = \sum_{i=0}^{n} \frac{(|y_i - \hat{y_i}|)}{C \cdot psnr(y_i, \hat{y_i})} \quad (1)$$

where C is a fixed coefficient; y and $\hat{y}$ are ground truth and predicted images. Optimization was performed using the Adam algorithm with a batch size of 32 and an initial learning rate of 1e-4, which decreased during training.

4.7. SJP-ISP

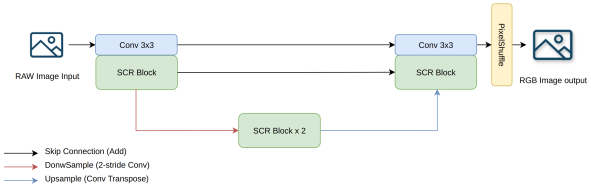


Figure 13. The U-Net-like architecture proposed by the SJP-ISP team.

The SJP-ISP team proposed a Simple yet Powerful Deep Neural ISP model (Fig. 13), emphasizing a minimalist design using only convolution layers for computational efficiency and hardware flexibility. Based on the U-Net [69] architecture, the model employs 2-stride convolutions for downsampling and transposed convolutions for upsampling, concluding with a PixelShuffle layer for a final 2x resolution increase. Each level within the U-Net structure consists of a Simple Convolution-ReLU (SCR) Block, which includes two 3×3 convolution layers with ReLU activations and a residual connection. This streamlined block design allows for potential replacement with reparameterizable blocks (e.g., RepVGG [16]) for platform-specific optimizations.

The training incorporated a multi-stage loss strategy and a joint denoising/demosaicing data augmentation pipeline. Initially, a combination of L_1 loss, PSNR loss, and a color loss based on the Gray World Assumption was used. After approximately 50 epochs, GAN loss [78] and VGG19 perceptual loss [55] were added. The final weighted loss function was:

$$Loss = \mathcal{L}_{L1} + \mathcal{L}_{PSNR} + 5 \cdot \mathcal{L}_{Color} + 0.005 \cdot \mathcal{L}_{GAN} + 1.0 \cdot \mathcal{L}_{Perceptual} \quad (2)$$

To enable joint learning, RAW domain data augmentation simulated low-light conditions and added realistic noise [7] according

to the following process:

$$
\begin{aligned}
F_{darken} &\sim \text{Uniform}(0.3, 0.7) \\
I_{darken} &= I \cdot (F_{darken} / \text{mean}(I)) \\
I_{noisy} &= I_{darken} + \text{noise}
\end{aligned} \tag{3}
$$

This augmentation strategy enhances robustness and noise reduction performance. The model was trained using the Adam optimizer with a learning rate of 5e-5.

4.8. D-Image

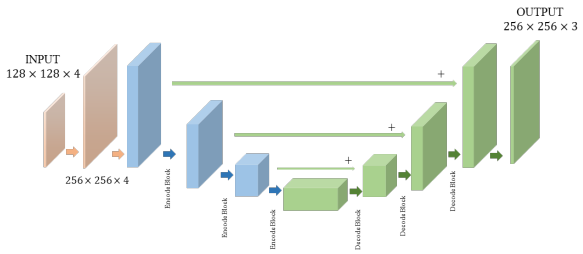


Figure 14. The U-Net-like architecture proposed by the D-Image team.

Team D-Image presented a Simplified Channel Attention-based U-net, drawing inspiration from U-Net [69] and NAFNet [10]. The data preprocessing involves nearest-neighbor up-sampling of the input, followed by channel expansion via separable convolution. The core network (Fig. 14) follows a U-Net structure with three encoder and three decoder levels linked by skip connections. Each level employs identical building blocks (detailed in Fig. 15) which feature Normalization Layers, Conv2d layers, and SimpleGate activations (splitting features along the channel dimension and multiplying the halves). These blocks also incorporate a simplified channel attention (SimpleCA) mechanism, consisting of adaptive global average pooling multiplied by the feature map, within one of their internal parallel paths. Residual connections are also utilized within the block structure. Final postprocessing involves reducing the channel count to 3 using separable convolution.

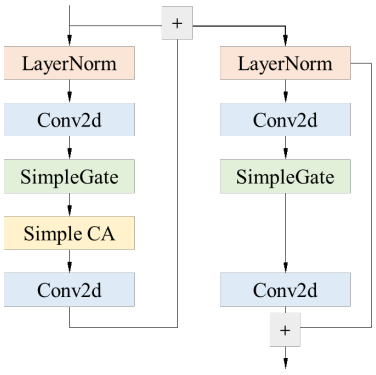


Figure 15. The Channel Attention-based block structure.

The training employs MSE loss with the Adam optimizer. The learning rate is decayed from 1e-3 to 1e-5 using cosine annealing. With a batch size of 32, each epoch consists of 500 iterations, and the model is trained for a total of 2000 epochs to achieve the final performance.

4.9. oteam

Team oteam developed an encoder-decoder architecture featuring symmetric skip connections, specifically designed for restoring RGB images from low-light RAW inputs. The model consists of four distinct components: (1) a Dynamic Pruning Encoder that performs hierarchical feature extraction using Conv-LeakyReLU blocks, max-pooling, and an adaptive channel scaling mechanism dependent on a pruning factor α; (2) a Hybrid Attention Bottleneck that integrates depthwise-separable convolutions (within a KBBlock module featuring channel attention) and a Vision Transformer (ViT) block utilizing multi-head self-attention; (3) a Multi-scale Fusion Decoder employing transposed convolutions for upsampling and concatenating features from corresponding encoder stages via skip connections; and (4) a Subpixel Reconstruction Head using PixelShuffle to generate the final RGB output while reducing artifacts.

For training, the team utilized a multi-scale Laplacian pyramid image loss function ($\mathcal{L}_{Lap}$) to measure cross-scale structural differences. The k-th level of the Laplacian pyramid, L_k, is defined as the difference between the k-th Gaussian pyramid level G_k and the upsampled $(k+1)$-th level:

$$
L_k = G_k - \uparrow (G_{k+1} * g_\sigma) \tag{4}
$$

where $G_{k+1} = \downarrow (G_k)$ represents downsampling, $\uparrow(\cdot)$ is bilinear upsampling, and g_σ is a Gaussian filter. The total loss combines the L1 norms of these residuals across K pyramid levels (typically $K = 5$):

$$
\mathcal{L}_{Lap} = \sum_{k=0}^{K-1} \|L_k(I) - L_k(T)\|_1 \tag{5}
$$

This loss function effectively captures differences in high-frequency details like edges and textures across multiple scales. Comprehensive data augmentation included random Bayer pattern extraction, 8 modes of geometric transformations (flips/transpositions), and dynamic patch sampling (128-256 pixels). Inference employed Test-Time Augmentation (TTA) by averaging predictions from four input orientations.

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 Super-Resolution: [8, 27, 63, 74]
- Learned End-to-End ISP: [29, 33, 42, 43, 80]
- RGB Image Enhancement: [23–25, 27]
- Bokeh Effect Rendering: [28, 31, 32]
- Image Denoising: [1, 2, 17]

Acknowledgements

We thank AI Witchlabs, ETH Zürich (Computer Vision Lab), and the University of Würzburg (Computer Vision Lab), the organizers

and sponsors of this Mobile AI 2025 challenge.

A. Teams and Affiliations

Mobile AI 2025 Team

Title:
Mobile AI 2025 Learned Smartphone ISP Challenge
Members:
Andrey Ignatov[1,3] *(andrey@vision.ee.ethz.ch)*, Georgy Perevozchikov[2] *(georgii.perevozchikov@uni-wuerzburg.de)*, Radu Timofte[2,3] *(radu.timofte@uni-wuerzburg.de)*
Affiliations:
[1] Computer Vision Lab, ETH Zurich, Switzerland
[2] Computer Vision Lab, University of Würzburg, Germany
[3] AI Witchlabs, Switzerland

AwesomeNeural

Title:
An Optimized Architecture Based on SYENet [18]
Members:
Cheng Li (1157456080@qq.com), Lian Liu, Jun Cao, Heng Sun
Affiliations:
Xiaomi Inc.

DHTXSF-Lab

Title:
DBF-Net: Diverse Branch Fusion with Channel-Scale for Enhanced Image Reconstruction
Members:
Wu Pan (269056278@qq.com), Song Wang, KeQiang Yu, Shuo Liu, HongQin He, ZhenHao Dong, JianKe Chen, DeJun Hao
Affiliations:
Dahua Technology

AVC2

Title:
MobileIE: Towards Efficient Real-Time Image Enhancement for Mobile Devices
Members:
Hailong Yan (yanhailong@std.uestc.edu.cn), Ao Li, Xiangtao Zhang, Zhe Liu, Ce Zhu, Le Zhang
Affiliations:
School of Information and Communication Engineering, University of Electronic Science and Technology of China (UESTC)

SHBD

Title:
Multiparallel nonlinear activation functions for image reconstruction
Members:
Keqiang Yu (19906782792@163.com), Tingniao Wang, Xiaoqing Zhou, Dong Zhang, Chunxia Lei, Jianke Chen, Dejun Hao, Jianguang He
Affiliations:
Dahua Technology

DHTSDX

Title:
RepSepConv with Adaptive Channel Scaling
Members:
Shuo Liu (shuo_liu1@outlook.com)
Affiliations:
Dahua Technology

UAIC ISP

Title:
Learned Lightweight Smartphone ISP with Unpaired Data [3]
Members:
Andrei Arhire (andreiarhire2708@gmail.com), Radu Timofte
Affiliations:
Faculty of Computer Science, Alexandru Ioan Cuza University of Iasi (UAIC)

SJP-ISP

Title:
SJP-ISP: Simple Joint Processing ISP with Degradation Pipeline
Members:
Junpyo Seo (seojun0602@gmail.com)
Affiliations:
Handong Global University

D-Image

Title:
Simplified Channel Attention-based U-net
Members:
Zhenhao Dong (henry_dzhh@outlook.com), Jianke Chen, Fen Xie, Xiuzhi Fang
Affiliations:
Dahua Technology

oteam

Title:
RAW2RGB
Members:
Chen Wu (wc915495936@live.com), Zhansheng Wang, Pengbo Zhang, Jiazi Huang

References

[1] 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. 8

[2] 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. 8

[3] Andrei Arhire and Radu Timofte. Learned lightweight smartphone isp with unpaired data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2025. 6, 9

[4] Yash Bhalgat, Jinwon Lee, Markus Nagel, Tijmen Blankevoort, and Nojun Kwak. Lsq+: Improving low-bit quantization through learnable offsets and better initialization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 696–697, 2020. 2

[5] Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. Understanding and overcoming the challenges of efficient transformer quantization. *arXiv preprint arXiv:2109.12948*, 2021. 2

[6] Yelysei Bondarenko, Riccardo Del Chiaro, and Markus Nagel. Low-rank quantization-aware training for llms. *arXiv preprint arXiv:2406.06385*, 2024. 2

[7] Tim Brooks, Ben Mildenhall, Tianfan Xue, Jiawen Chen, Dillon Sharlet, and Jonathan T Barron. Unprocessing images for learned raw denoising. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11036–11045, 2019. 7

[8] 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. 8

[9] 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. 2

[10] Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration. In *European conference on computer vision*, pages 17–33. Springer, 2022. 5, 8

[11] 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. 2

[12] Krishna Teja Chitty-Venkata and Arun K Somani. Neural architecture search survey: A hardware perspective. *ACM Computing Surveys*, 55(4):1–36, 2022. 2

[13] Grace Chu, Okan Arikan, Gabriel Bender, Weijun Wang, Achille Brighton, Pieter-Jan Kindermans, Hanxiao Liu, Berkin Akin, Suyog Gupta, and Andrew Howard. Discovering multi-hardware mobile models via architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3022–3031, 2021. 2

[14] Linhui Dai, Xiaohong Liu, Chengqi Li, and Jun Chen. Awnet: Attentive wavelet network for image isp. *arXiv preprint arXiv:2008.09228*, 2020. 1

[15] Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Diverse branch block: Building a convolution as an inception-like unit. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10886–10895, 2021. 5

[16] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13733–13742, 2021. 7

[17] Roman Flepp, Andrey Ignatov, Radu Timofte, and Luc Van Gool. Real-world mobile image denoising dataset with efficient baselines. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22368–22377, 2024. 8

[18] Weiran Gou, Ziyao Yi, Yan Xiang, Shaoqing Li, Zibin Liu, Dehui Kong, and Ke Xu. Syenet: A simple yet effective network for multiple low-level vision tasks with real-time performance on mobile device. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12182–12195, 2023. 5, 9

[19] Benjamin Hawks, Javier Duarte, Nicholas J Fraser, Alessandro Pappalardo, Nhan Tran, and Yaman Umuroglu. Ps and qs: Quantization-aware pruning for efficient low latency neural network inference. *Frontiers in Artificial Intelligence*, 4: 676564, 2021. 2

[20] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 5

[21] 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. 2

[22] Mu Hu, Junyi Feng, Jiashen Hua, Baisheng Lai, Jianqiang Huang, Xiaojin Gong, and Xian-Sheng Hua. Online convolutional re-parameterization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 568–577, 2022. 5, 7

[23] 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. 8

[24] 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.

[25] 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. 8

[26] 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. 2, 3

[27] 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. 8

[28] 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. 8

[29] 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. 1, 8

[30] 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. 2, 3

[31] 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. 8

[32] 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. 8

[33] 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. 1, 8

[34] 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, 4, 5

[35] Andrey Ignatov, Kim Byeoung-su, and Radu Timofte. Fast camera image denoising on mobile gpus 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

[36] 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, 4, 6

[37] 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

[38] 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

[39] 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

[40] 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

[41] Andrey Ignatov, Grigory Malivenko, Radu Timofte, Lukasz Treszczotko, Xin Chang, Piotr Ksiazek, Michal Lopuszynski, Maciej Pioro, Rafal Rudnicki, Maciej Smyl, et al. Efficient single-image depth estimation on mobile devices, mobile ai & aim 2022 challenge: report. In *European Conference on Computer Vision*, pages 71–91. Springer, 2022. 2

[42] Andrey Ignatov, Grigory Malivenko, Radu Timofte, Yu Tseng, Yu-Syuan Xu, Po-Hsiang Yu, Cheng-Ming Chiang, Hsien-Kai Kuo, Min-Hung Chen, Chia-Ming Cheng, and Luc Van Gool. Pynet-v2 mobile: Efficient on-device photo processing with neural networks. In *2021 26th International Conference on Pattern Recognition (ICPR)*. IEEE, 2022. 1, 2, 4, 5, 8

[43] Andrey Ignatov, Anastasia Sycheva, Radu Timofte, Yu Tseng, Yu-Syuan Xu, Po-Hsiang Yu, Cheng-Ming Chiang, Hsien-Kai Kuo, Min-Hung Chen, Chia-Ming Cheng, and Luc Van Gool. Microisp: Processing 32mp photos on mobile devices with deep learning. In *European Conference on Computer Vision*, 2022. 1, 2, 4, 8

[44] Andrey Ignatov, Radu Timofte, Cheng-Ming Chiang, Hsien-Kai Kuo, Yu-Syuan Xu, Man-Yu Lee, Allen Lu, Chia-Ming Cheng, Chih-Cheng Chen, Jia-Ying Yong, et al. Power efficient video super-resolution on mobile npus with deep learning, mobile ai & aim 2022 challenge: Report. In *European Conference on Computer Vision*, pages 130–152. Springer, 2022. 2

[45] Andrey Ignatov, Radu Timofte, Shuai Liu, Chaoyu Feng, Furui Bai, Xiaotao Wang, Lei Lei, Ziyao Yi, Yan Xiang, Zibin Liu, et al. Learned smartphone isp on mobile gpus with deep learning, mobile ai & aim 2022 challenge: report. In *European Conference on Computer Vision*, pages 44–70. Springer, 2022. 2, 5, 6, 7

[46] Andrey Ignatov, Radu Timofte, Jin Zhang, Feng Zhang, Gaocheng Yu, Zhe Ma, Hongbin Wang, Minsu Kwon, Haotian Qian, Wentao Tong, et al. Realistic bokeh effect rendering on mobile gpus, mobile ai & aim 2022 challenge: re-

port. In *European Conference on Computer Vision*, pages 153–173. Springer, 2022. 2

[47] Andrey Ignatov, Georgy Perevozchikov, and Radu Timofte. Rgb photo enhancement on mobile gpus, mobile ai 2025 challenge: Report. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2025. 2

[48] Andrey Ignatov, Georgy Perevozchikov, and Radu Timofte. Quantized image super-resolution on mobile npus, mobile ai 2025 challenge: Report. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2025. 2

[49] Dmitry Ignatov and Andrey Ignatov. Controlling information capacity of binary neural network. *Pattern Recognition Letters*, 138:276–281, 2020. 2

[50] 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. 2

[51] 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. 2

[52] Byung-Hoon Kim, Joonyoung Song, Jong Chul Ye, and Jae-Hyun Baek. Pynet-ca: enhanced pynet with channel attention for end-to-end mobile image signal processing. In *European Conference on Computer Vision*, pages 202–212. Springer, 2020. 1

[53] Jangho Kim, Simyung Chang, and Nojun Kwak. Pqk: model compression via pruning, quantization, and knowledge distillation. *arXiv preprint arXiv:2106.14681*, 2021. 2

[54] Andrey Kuzmin, Markus Nagel, Mart Van Baalen, Arash Behboodi, and Tijmen Blankevoort. Pruning vs quantization: Which is better? *Advances in neural information processing systems*, 36:62414–62427, 2023. 2

[55] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017. 7

[56] Juhyun Lee, Nikolay Chirkov, Ekaterina Ignasheva, Yury Pisarchyk, Mogan Shieh, Fabio Riccardi, Raman Sarokin, Andrei Kulik, and Matthias Grundmann. On-device neural net inference with mobile gpus. *arXiv preprint arXiv:1907.01989*, 2019. 3

[57] Fei Li, Wenbo Hou, and Peng Jia. Rmfa-net: A neural isp for real raw to rgb image reconstruction. *arXiv preprint arXiv:2406.11469*, 2024. 6, 7

[58] 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. 2

[59] Yuhang Li, Mingzhu Shen, Jian Ma, Yan Ren, Mingxin Zhao, Qi Zhang, Ruihao Gong, Fengwei Yu, and Junjie Yan. Mqbench: Towards reproducible and deployable model quantization benchmark. *arXiv preprint arXiv:2111.03759*, 2021. 2

[60] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Guangxuan Xiao, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *GetMobile: Mobile Computing and Communications*, 28(4): 12–17, 2025. 2

[61] 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. 2

[62] 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. 2

[63] 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. 8

[64] Keith G Mills, Fred X Han, Jialin Zhang, Seyed Saeed Changiz Rezaei, Fabian Chudak, Wei Lu, Shuo Lian, Shangling Jui, and Di Niu. Profiling neural blocks and design spaces for mobile neural architecture search. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 4026–4035, 2021. 2

[65] Markus Nagel, Marios Fournarakis, Rana Ali Amjad, Yelysei Bondarenko, Mart Van Baalen, and Tijmen Blankevoort. A white paper on neural network quantization. *arXiv preprint arXiv:2106.08295*, 2021. 2

[66] 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. 2

[67] LiteRT overview. `https://ai.google.dev/edge/litert`, 2025. 2

[68] Angeline Pouget, Sidharth Ramesh, Maximilian Giang, Ramithan Chandrapalan, Toni Tanner, Moritz Prussing, Radu Timofte, and Andrey Ignatov. Fast and accurate camera scene detection on smartphones. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2569–2580, 2021. 2

[69] 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. 7, 8

[70] Sangeetha Siddegowda, Marios Fournarakis, Markus Nagel, Tijmen Blankevoort, Chirag Patel, and Abhijit Khobare. Neural network quantization with ai model efficiency toolkit (aimet). *arXiv preprint arXiv:2201.08442*, 2022. 2

[71] Jose Ivson S Silva, Gabriel G Carvalho, Marcel Santana Santos, Diego JC Santiago, Lucas Pontes de Albuquerque, Jorge

F Puig Battle, Gabriel M da Costa, and Tsang Ing Ren. A deep learning approach to mobile camera image signal processing. In *Anais Estendidos do XXXIII Conference on Graphics, Patterns and Images*, pages 225–231. SBC, 2020. 1

[72] Fuwen Tan, Royson Lee, Shell Xu Hu, Sourav Bhattacharya, Timothy Hospedales, Georgios Tzimiropoulos, Brais Martinez, et al. Mobilequant: Mobile-friendly quantization for on-device language models. *arXiv preprint arXiv:2408.13933*, 2024. 2

[73] 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. 2

[74] 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. 8

[75] Prune Truong, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning accurate dense correspondences and when to trust them. *arXiv preprint arXiv:2101.01710*, 2021. 3

[76] 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. 2

[77] 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. 2

[78] Xintao Wang, Liangbin Xie, Chao Dong, and Ying Shan. Real-esrgan: Training real-world blind super-resolution with pure synthetic data. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1905–1914, 2021. 7

[79] Zhiyuan Wang and Haisheng Tan. Towards efficient inference on mobile device via pruning. In *2024 10th International Conference on Big Data Computing and Communications (BigCom)*, pages 26–33. IEEE, 2024. 2

[80] Daniel Wirzberger Raimundo, Andrey Ignatov, and Radu Timofte. Lan: Lightweight attention-based network for raw-to-rgb smartphone image processing. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 807–815. IEEE, 2022. 8

[81] 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. 2

[82] Yushu Wu, Yifan Gong, Pu Zhao, Yanyu Li, Zheng Zhan, Wei Niu, Hao Tang, Minghai Qin, Bin Ren, and Yanzhi Wang. Compiler-aware neural architecture search for on-mobile real-time super-resolution. In *European Conference on Computer Vision*, pages 92–111. Springer, 2022. 2

[83] Haitao Xu, Songwei Liu, Yuyang Xu, Shuai Wang, Jiashi Li, Chenqian Yan, Liangqiang Li, Lean Fu, Xin Pan, and Fangmin Chen. Sparsebytenn: A novel mobile inference acceleration framework based on fine-grained group sparsity. *arXiv preprint arXiv:2310.19509*, 2023. 2

[84] 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. 2

[85] Yutaka Yoshihama, Kenichi Yadani, and Shota Isobe. Hardware-aware zero-shot neural architecture search. In *2023 18th International Conference on Machine Vision and Applications (MVA)*, pages 1–5. IEEE, 2023. 2

[86] Zheng Zhan, Yifan Gong, Pu Zhao, Geng Yuan, Wei Niu, Yushu Wu, Tianyun Zhang, Malith Jayaweera, David Kaeli, Bin Ren, et al. Achieving on-mobile real-time super-resolution with neural architecture and pruning search. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4821–4831, 2021. 2

[87] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 7

[88] Yu-Jie Zheng, Si-Bao Chen, Chris HQ Ding, and Bin Luo. Model compression based on differentiable network channel pruning. *IEEE Transactions on Neural Networks and Learning Systems*, 34(12):10203–10212, 2022. 2

[89] Shaojie Zhuo, Hongyu Chen, Ramchalam Kinattinkara Ramakrishnan, Tommy Chen, Chen Feng, Yicheng Lin, Parker Zhang, and Liang Shen. An empirical study of low precision quantization for tinyml. *arXiv preprint arXiv:2203.05492*, 2022. 2