Real-Time Quantized Image Super-Resolution on Mobile NPUs, Mobile AI 2021 Challenge: Report

Andrey Ignatov Radu Timofte Maurizio Denna Abdel Younes Andrew Lek
Mustafa Ayazoglu Jie Liu Zongcai Du Jiaming Guo Xueyi Zhou Hao Jia
Youliang Yan Zexin Zhang Yixin Chen Yunbo Peng Yue Lin Xindong Zhang
Hui Zeng Kun Zeng Peirong Li Zhihuang Liu Shiqi Xue Shengpeng Wang

Abstract

Image super-resolution is one of the most popular computer vision problems with many important applications to mobile devices. While many solutions have been proposed for this task, they are usually not optimized even for common smartphone AI hardware, not to mention more constrained smart TV platforms that are often supporting INT8 inference only. 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 super-resolution solutions that can demonstrate a real-time performance on mobile or edge NPUs. For this, the participants were provided with the DIV2K dataset and trained quantized models to do an efficient 3X image upscaling. The runtime of all models was evaluated on the Synaptics VS680 Smart Home board with a dedicated NPU capable of accelerating quantized neural networks. The proposed solutions are fully compatible with all major mobile AI accelerators and are capable of reconstructing Full HD images under 40-60 ms while achieving high fidelity results. A detailed description of all models developed in the challenge is provided in this paper.

1. Introduction

Image super-resolution is a classical computer vision problem where the goal is to reconstruct the original image based on its downscaled version, adding the lost lost high frequencies and rich texture details. During the past years, this task has witnessed an increased popularity due to its direct application to telephoto image processing in smartphone cameras, low-resolution media data enhancement as well as to upscaling images and videos to the target high resolution of display panels. Numerous classical [36, 14, 46, 50, 51, 57, 59, 60, 16, 53] and deep learning-based [12, 11, 39, 41, 49, 52, 6, 44, 61, 31] approaches have been proposed for this task in the past. The biggest limitation of these methods is that they were primarily targeted at achieving high fidelity scores while not optimized for computational efficiency and mobile-related constraints, which is essential for this and other tasks related to image processing and enhancement [19, 20, 33] on mobile devices. In this challenge, we take one step further in solving this problem by using a popular DIV2K [3] image super-resolution dataset and by imposing 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 [29, 27]. 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. 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 [8, 23, 40, 42, 45], 16-bit / 8-bit [8, 38, 37, 58] and low-bit [7, 54, 34, 43] quantization, device- or NPU-specific adaptations, platform-aware neural architecture search [15, 47, 56, 55], *etc*.

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

* Andrey Ignatov, Radu Timofte, Maurizio Denna and Abdel Younes are the Mobile AI 2021 challenge organizers *(andrey@vision.ee.ethz.ch, radu.timofte@vision.ee.ethz.ch, maurizio.denna@synaptics.com, abdel.younes@synaptics.com)*. 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 a 3X bicubically upscaled image and the target DIV2K [3] photo.

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 the DIV2K [3] dataset containing diverse 2K resolution RGB images used to train their models using a downscaling factor of 3. More importantly, since many mobile and smart TV platforms can accelerate only INT8 models, all submitted solutions had to be fully-quantized. Within the challenge, the participants were evaluating the runtime and tuning their models on the Synaptics Dolphin platform featuring a dedicated NPU that can efficiently accelerate INT8 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 [48], 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 [9].

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

- Learned Smartphone ISP on Mobile NPUs [18]
- Real Image Denoising on Mobile GPUs [17]
- Quantized Image Super-Resolution on Edge SoC NPUs
- Real-Time Video Super-Resolution on Mobile GPUs [25]
- Single-Image Depth Estimation on Mobile Devices [21]
- Quantized Camera Scene Detection on Smartphones [22]
- 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;
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

In this challenge, the participants were proposed to work with the popular DIV2K [3] dataset. It consists from 1000 divers 2K resolution RGB images: 800 are used for training, 100 for validation and 100 for testing purposes. The images are of high quality both aesthetically and in the terms of small amounts of noise and other corruptions (like blur and color shifts). All images were manually collected and have 2K pixels on at least one of the axes (vertical or horizontal). DIV2K covers a large diversity of contents, from people, handmade objects and environments (cities), to flora and fauna and natural sceneries, including underwater. An example set of images is demonstrated 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 [27, 29] that allows to load any custom TensorFlow Lite model and run it on any Android device with all supported

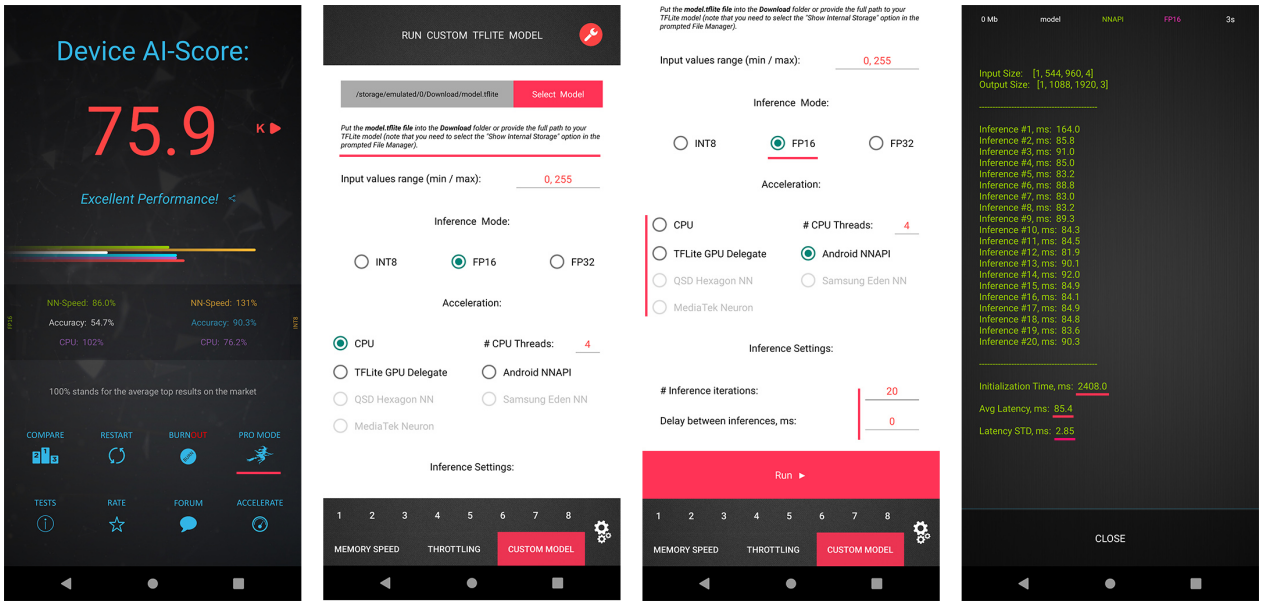

Figure 2. Loading and running custom TensorFlow Lite models with AI Benchmark application. The currently supported acceleration options include Android NNAPI, TFLite GPU, Hexagon NN, Samsung Eden and MediaTek Neuron delegates as well as CPU inference through TFLite or XNNPACK backends. The latest app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

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.
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 *Synaptics VS680 Edge AI SoC* [35] Evaluation Kit as our target runtime evaluation platform. The VS680 Edge AI SoC is integrated into Smart Home solution and it features a powerful NPU designed by *VeriSilicon* and capable of accelerating quantized models (up to 7 TOPS). It supports Android and can perform NN inference through NNAPI, demonstrating INT8 AI Benchmark scores that are close to the ones of mid-range smartphone chipsets. Within the challenge, the participants were able to upload their TFLite models to an external server and get feedback regarding the speed of their model: the inference time of their solution on the above mentioned NPU or an error log if the network contained incompatible operations and/or improper quantization. Participants' models were parsed and accelerated using Synaptics' TFLite delegate that can dynamically parse and map a given model's higher level representation of neural network layers and operations to its equivalent internal binary representation that will be optimized for efficient integer only execution on the VS680's NPU. The same setup was also used for the final runtime evaluation. The participants were additionally provided with a list of ops supported by this board and model optimization guidance in order to fully utilize the NPU's convolution and tensor processing resources.

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 plat-

[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↑ INT8 Model	SSIM↑ INT8 Model	Δ PSNR FP32 → INT8	Δ SSIM Acc. Drop	Runtime, ms ↓ CPU	Runtime, ms ↓ NPU	Speed-Up	Final Score
Aselsan Research	deepernewbie	Keras / TensorFlow	67	29.58	0.86	0.18	0.0093	1278	44.85	28.5	**51.02**
Noah_TerminalVision	JeremieG	Keras / TensorFlow	109	29.41	0.8537	0.33	0.0142	668	**38.32**	17.4	47.18
ALONG	richlaji	TensorFlow	30	29.52	0.8607	N.A.	N.A.	951	62.25	15.3	33.82
A+ regression [51]	Baseline			29.32	0.8520	-	-	-	-	-	-
EmbededAI	xindongzhang	PyTorch / TensorFlow	82	28.82	0.8428	0.15	0.0119	1224	76.61	16	10.41
mju_gogogo	mju_gogogo	Keras / TensorFlow	940	28.92	0.8486	N.A.	N.A.	Failed	718	-	1.28
Bicubic Upscaling	Baseline			28.26	0.8277	-	-	-	-	-	-
221B	masanshu	TensorFlow	175	25.44	0.729	4.03	0.1337	Failed	238.43	-	0.03
svnit_ntnu	kalpesh_svnit	TensorFlow	8	19.3	0.7061	9.61	0.1442	1947	78.84	24.7	0.00
CVML	vishalchudasama	TensorFlow	10	19.5	0.7462	9.45	0.1049	1772	90.20	19.6	0.00
TieGuoDun Team	ShuaiqiXiaopangzhi	TensorFlow	636	16.19	0.6654	13.28	0.1991	Failed	913.96	-	0.00
MCG *	*TinyJie*	TensorFlow	53	**29.87**	**0.8686**	N.A.	N.A.	998	36.89	27	**92.72**

Table 1. Mobile AI 2021 Real-Time Image Super-Resolution challenge results and final rankings. During the runtime measurements, the models were performing image upscaling from 640×360 to 1920×1080 pixels. Δ PSNR and Δ SSIM values correspond to accuracy loss measured in comparison to the original floating-point network. Team *Aselsan Research* is the challenge winner. * The presented solution from *TinyJie* team was submitted after the official challenge deadline.

form, 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 Synaptics VS680 board.

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 180 registered participants, 12 teams entered the final phase and submitted their results, TFLite models, codes, executables and factsheets. Table 1 summarizes the final challenge results and reports PSNR, SSIM and runtime numbers for the valid 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

The problem considered in this competition was very challenging as the solutions had to be both optimized for the target Smart Home platform and be fully quantized. While there exists many works proposing various weight quantization techniques, the majority of them are still using floating-point activations: in this case, the resulting models are not compatible with INT8 NPUs, and the entire idea of network quantization is lost as no benefits are obtained on real hardware. Therefore, in this challenge the participants had to perform full model quantization including inputs, weights, convolutions and activations – networks with floating-point ops were not accepted. As one can see (Fig. 1), only 6 teams were able to outperform a simple bicubic image upsampling baseline, in all other cases the accuracy loss after quantization was enormous, and the models were just producing corrupted outputs. One major issue met by the majority of challenge participants was to avoid using floating-point output dequantization: without this block, quantized outputs were often normalized incorrectly, resulting in wrong network's output values scaling. The easiest way to deal with this problem was to add a simple clipped *ReLU* layer on top of the model, then the outputs were mapped linearly to the [0, 255] interval. Applying quantized-aware training was also helping a lot in getting good fidelity results.

The majority of the proposed solutions demonstrated a very high efficiency, being able to upscale 640×360 pixel input images to Full HD resolution under 60-80 ms on the target Synaptics VS680 board. Since not all TFLite operations were equally optimized by the target platform, participants had to rely on a recommended set of ops for building their models in order to design a solution that would maximize NPU utilization. Team *Aselsan Research* is the challenge winner — the authors were able to achieve good fidelity and runtime values by using a relatively small model with grouped convolutions. Even better results were obtained by team *MCG*, though, unfortunately,

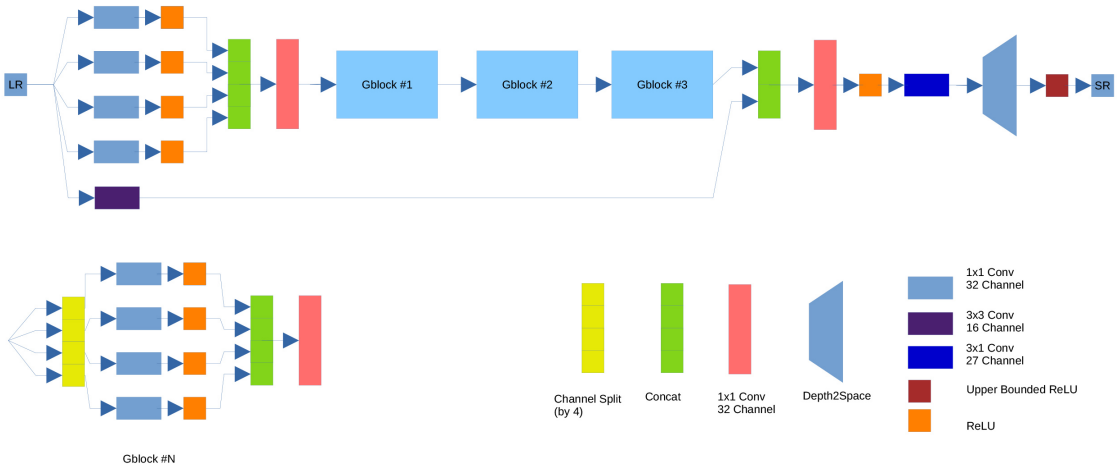

Figure 3. The model architecture proposed by Aselsan Research team.

it was able to solve all issues related to model quantization only after the end of the challenge. This team as well as *Noah_TerminalVision* and *ALONG* were using a similar idea of learning only SR residuals with convolutional or residual blocks. Quantized models submitted by *EmbededAI* and *mju_gogogo* demonstrated only a slight improvement over the baseline bicubic upscaling approach. The rest of the teams were not able to fight the accuracy drop resulted from model quantization, though their original floating-point networks were showing quite good fidelity scores.

At the beginning of the runtime validation phase, almost all submitted solutions were either crushing on the target Synaptics platform or demonstrating a runtime of several seconds when tested on the target resolution images. It took a large number of iterations for the majority of teams to come up with solutions that can be accelerated efficiently on the considered NPU, though many models were very light and demonstrated good speed on desktop CPUs and GPUs from the very beginning. 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 IoT and mobile 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-Time Image Super-Resolution challenge.

4.1. Aselsan Research

The architecture proposed by Aselsan Research is presented in Fig. 3. The major building block (Gblock) of this model is based on the concept of grouped convolutions: the input feature maps are split into 4 parts and fed to separate convolutions (working in parallel) to decrease the RAM consumption and computational costs. The authors emphasize that though replacing the standard convolutional layers with separable convolutions results in better runtime, this also leads to a large accuracy drop after performing model quantization, thus they were not used in the final architecture. An additional skip connection was added to improve the fidelity results of the proposed solution, no input data normalization was used to increase the speed of the model.

The network was trained on 32×32 pixel input images with a batch size of 16. *Charbonnier* loss function was minimized using Adam optimizer with a dynamic learning rate ranging from $25e-4$ to $1e-4$. Model quantization was performed with TensorFlow's standard post-training quantization utilities, clipped *ReLU* was added on top of the model to avoid incorrect output normalization. A more detailed description of the model, design choices and training procedure is provided in [5].

4.2. MCG

Team MCG proposed an anchor-based CNN (Fig. 4) for the considered problem. The main idea behind this architecture is to learn only the residual part of SR image. If we remove all convolutional layers from this model, then the workflow would be as follows: the input image is stacked $3\times3 = 9$ times (where 3 is the upscaling factor) and then reshaped by the *depth-to-space* layer to the target resolution. The resulting image will have the same size as the tar-

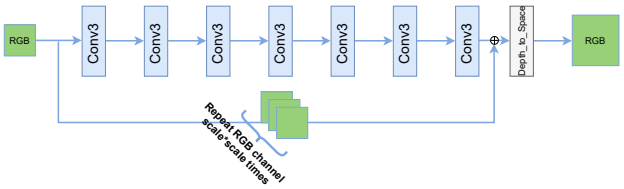

Figure 4. Anchor-based CNN proposed by MCG team.

get SR one — resizing is achieved by repeating each pixel value 9 times. The "removed" convolutional block is therefore learning the difference between the low-resolution and SR photos, which is added to the input image before the *depth-to-space* layer. This block consists of five 3×3 convolutional layers followed by *ReLU* activations, and one additional conv layer on top of them.

The network was trained with a batch size of 16 on 64×64 pixel input images augmented by random flipping and rotation. L_1 loss was used as a target metric, model parameters were optimized for 1000 epochs using Adam with a learning rate initialized at $1e-3$ and decreases by half every 200 epochs. Quantized-aware training as well as post-training quantization were applied to get an accurate INT8 model, clipped *ReLU* was added on top of the network to avoid incorrect output normalization. A detailed description of the proposed method is also provided in [13].

4.3. Noah_TerminalVision

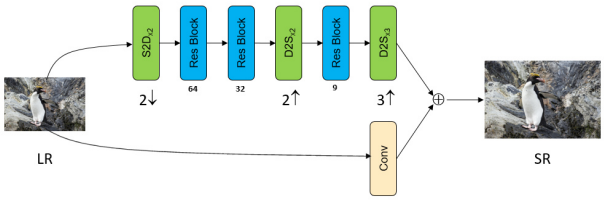

Figure 5. TinySRNet model designed by Noah_TerminalVision.

Team Noah_TerminalVision developed a small TinySRNet model demonstrated in Fig. 5. This network contains three residual blocks (each consisting of two convolutions), space-to-depth (S2D), depth-to-space (D2S) and one residual convolutional layer. The authors especially emphasize the importance of the residual block which helps a lot in maintaining good accuracy after model quantization. The network was trained to minimize L_1 loss, its parameters were optimized using Adam for one million iterations with a cyclic learning rate starting from $5e-4$ and decreased to $1e-6$ each 200K iterations. Quantized-aware training was applied to improve the accuracy of the resulting INT8 model.

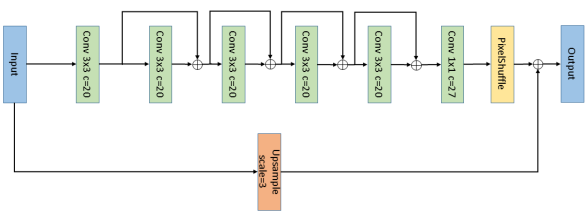

Figure 6. The model architecture proposed by ALONG team.

4.4. ALONG

The architecture developed by team ALONG (Fig. 6) is very similar to the previous solution, the major difference consists in doing all processing on the original scale and using nearest neighbor upsampling instead of convolution in the residual block connecting the input and output layers. The model was first trained to minimize L_1 loss function and then fine-tuned with L_2 loss. 128×128px input patches augmented with random flips and rotations were used during the training, model parameters were optimized using Adam with an initial learning rate of $2e-4$ halved every 200K iterations. Quantized-aware training was applied to improve the accuracy of the resulting INT8 model.

4.5. EmbededAI

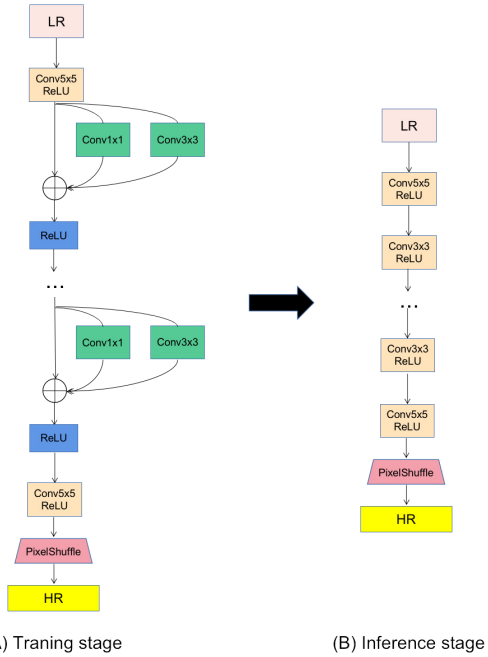

Figure 7. Network topology of the PRPSR CNN developed by EmbededAI team during training and inference stages.

Team EmbededAI presented a Plain Re-Parameterizable Convolutions for Super Resolution (PRPSR) model for the considered task. This network is designed using a pure con-

volutional topology: the input low-resolution image is fed to 5×5 convolutional layer performing feature extraction, followed by five 3×3 convolutions, one 3×3 convolutional and one pixel-shuffle layer reshaping the output to the target resolution (Fig. 7, right image). No residual blocks, skip connections, up-sampling or even addition operations are used in the model to improve its speed. During the training stage, each single convolutional layer is decoupled into a group of three operations:

$$y = x + conv_{1\times1}(x) + conv_{3\times3}(x),$$

followed by *ReLU* activations (Fig. 7, left image). After the end of the training, the parameters of each convolutional group are folded to get a single convolution [10].

The network was trained with a batch size of 32 on 64×64 pixel input images. L_1 loss was used as a target metric, model parameters were optimized for 600 epochs using Adam with an initial learning rate of $5e-4$ decreases by half every 200 epochs. Model quantization was performed with TensorFlow's post-training quantization tools, clipped *ReLU* was used after the last convolutional layer to avoid incorrect outputs normalization.

4.6. mju_gogogo

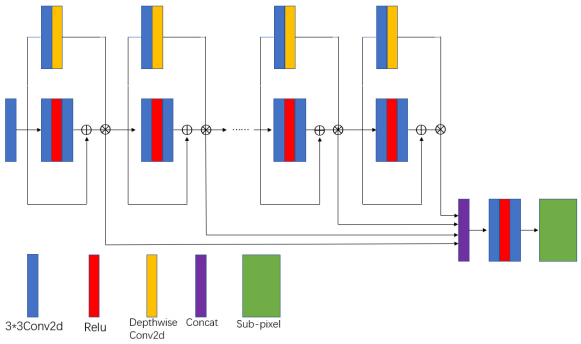

Figure 8. EDSR-based model with attention blocks proposed by mju_gogogo team.

The model architecture proposed by mju_gogogo team is presented in Fig. 8. The model is based on the EDSR [41] design with additional spatial attention blocks. The network is relatively large — it consists of six residual blocks, each one multiplied by the outputs of the corresponding attention unit, one 1×1 convolutional layer reducing the number of filters from 384 to 64, one global skip connection, three 3×3 convolutions and one *depth-to-space* layer. The model was first trained to minimize the *mean absolute error (MAE)* with Adam optimizer for 750 epochs, and then fine-tuned using quantized-aware training and the *MSE* loss for another 75 epochs.

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: [31, 44, 6, 52]
- Learned End-to-End ISP: [28, 32]
- Perceptual Image Enhancement: [31, 26]
- Bokeh Effect Rendering: [24, 30]
- Image Denoising: [1, 2]

Acknowledgements

We thank Synaptics Inc., 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 Image Super-Resolution Challenge
Members:
Andrey Ignatov[1,4] *(andrey@vision.ee.ethz.ch)*, Radu Timofte[1,4] *(radu.timofte@vision.ee.ethz.ch)*, Maurizio Denna[2] *(maurizio.denna@synaptics.com)*, Abdel Younes[2] *(abdel.younes@synaptics.com)*, Andrew Lek[3] *(andrew.lek@synaptics.com)*
Affiliations:
[1] Computer Vision Lab, ETH Zurich, Switzerland
[2] Synaptics Europe, Switzerland
[3] Synaptics HQ, San Jose, California, USA
[4] AI Witchlabs, Switzerland

Aselsan Research

Title:
Extremely Lightweight Quantization Robust Real-Time Single-Image Super Resolution for Mobile Devices [5]
Members:
Mustafa Ayazoglu (mayazoglu@aselsan.com.tr)
Affiliations:
Aselsan Corporation, Turkey
https://www.aselsan.com.tr/

MCG

Title:
Anchor-based Net for Mobile Image Super-Resolution [13]

Members:
Jie Liu (jieliu@smail.nju.edu.cn), Zongcai Du
Affiliations:
Nanjing University, China

Noah_TerminalVision

Title:
TinySRNet for Real-Time Image Super-Resolution
Members:
Jiaming Guo (guojiaming5@huawei.com), Xueyi Zhou, Hao Jia, Youliang Yan
Affiliations:
Huawei Technologies Co., Ltd, China

ALONG

Title:
FastSR: Solution for Real-Time Image Super-Resolution Challenge
Members:
Zexin Zhang (m15622188336@163.com), Yixin Chen, Yunbo Peng, Yue Lin
Affiliations:
Netease Games AI Lab, China

EmbededAI

Title:
Plain Re-parameterizable Convolutions for Efficient Super Resolution
Members:
Xindong Zhang (xindongzhang@foxmail.com), Hui Zeng
Affiliations:
The Hong Kong Polytechnic University, Hong Kong

mju_gogogo

Title:
EDSR-based Model with Attention Blocks for Image Super-Resolution
Members:
Kun Zeng (zengkun301@aliyun.com), Peirong Li, Zhihuang Liu, Shiqi Xue, Shengpeng Wang
Affiliations:
Minjiang University, 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. 7

[2] 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. 7

[3] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 126–135, 2017. 1, 2

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

[5] Mustafa Ayazoglu. Extremely lightweight quantization robust real-time single-image super resolution for mobile devices. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2021. 5, 7

[6] 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. 1, 7

[7] 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

[8] 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

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

[10] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. *arXiv preprint arXiv:2101.03697*, 2021. 7

[11] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *European conference on computer vision*, pages 184–199. Springer, 2014. 1

[12] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2015. 1

[13] Zongcai Du, Jie Liu, Jie Tang, and Gangshan Wu. Anchor-based plain net for mobile image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2021. 6, 7

[14] William T Freeman, Thouis R Jones, and Egon C Pasztor. Example-based super-resolution. *IEEE Computer graphics and Applications*, 22(2):56–65, 2002. 1

[15] 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

[16] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5197–5206, 2015. 1

[17] 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

[18] 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

[19] 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

[20] 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

[21] 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

[22] 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

[23] 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

[24] 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

[25] 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

[26] 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

[27] 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, 2

[28] 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

[29] 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, 2

[30] 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

[31] 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. 1, 7

[32] 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

[33] 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

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

[35] Synaptics Inc. https://www.synaptics.com/technology/edge-computing. 3

[36] Michal Irani and Shmuel Peleg. Improving resolution by image registration. *CVGIP: Graphical models and image processing*, 53(3):231–239, 1991. 1

[37] 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

[38] Sambhav R Jain, Albert Gural, Michael Wu, and Chris H Dick. Trained quantization thresholds for accurate and effi-

cient fixed-point inference of deep neural networks. *arXiv preprint arXiv:1903.08066*, 2019. 1

[39] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1646–1654, 2016. 1

[40] 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

[41] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017. 1, 7

[42] 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

[43] 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

[44] 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. 1, 7

[45] 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

[46] Sung Cheol Park, Min Kyu Park, and Moon Gi Kang. Super-resolution image reconstruction: a technical overview. *IEEE signal processing magazine*, 20(3):21–36, 2003. 1

[47] 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

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

[49] Radu Timofte, Eirikur Agustsson, Luc Van Gool, Ming-Hsuan Yang, and Lei Zhang. Ntire 2017 challenge on single image super-resolution: Methods and results. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 114–125, 2017. 1

[50] Radu Timofte, Vincent De Smet, and Luc Van Gool. Anchored neighborhood regression for fast example-based super-resolution. In *Proceedings of the IEEE international conference on computer vision*, pages 1920–1927, 2013. 1

[51] Radu Timofte, Vincent De Smet, and Luc Van Gool. A+: Adjusted anchored neighborhood regression for fast super-resolution. In *Asian conference on computer vision*, pages 111–126. Springer, 2014. 1, 4

[52] 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. 1, 7

[53] Radu Timofte, Rasmus Rothe, and Luc Van Gool. Seven ways to improve example-based single image super resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1865–1873, 2016. 1

[54] 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

[55] 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

[56] 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

[57] Chih-Yuan Yang and Ming-Hsuan Yang. Fast direct super-resolution by simple functions. In *Proceedings of the IEEE international conference on computer vision*, pages 561–568, 2013. 1

[58] 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

[59] Jianchao Yang, John Wright, Thomas Huang, and Yi Ma. Image super-resolution as sparse representation of raw image patches. In *2008 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2008. 1

[60] Jianchao Yang, John Wright, Thomas S Huang, and Yi Ma. Image super-resolution via sparse representation. *IEEE transactions on image processing*, 19(11):2861–2873, 2010. 1

[61] Kai Zhang, Shuhang Gu, and Radu Timofte. Ntire 2020 challenge on perceptual extreme super-resolution: Methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 492–493, 2020. 1