Extremely Lightweight Quantization Robust Real-Time Single-Image Super Resolution for Mobile Devices

Mustafa Ayazoglu
Aselsan Research
Ankara, Turkey
mayazoglu@aselsan.com.tr

Abstract

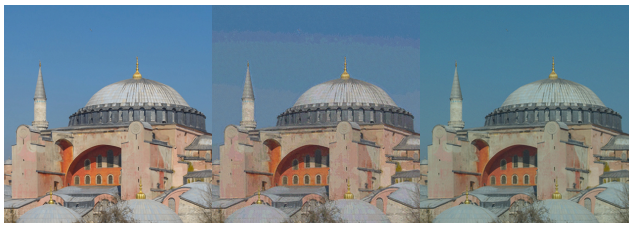
*Single-Image Super Resolution (SISR) is a classical computer vision problem and it has been studied for over decades. With the recent success of deep learning methods, recent work on SISR focuses solutions with deep learning methodologies and achieves state-of-the-art results. However most of the state-of-the-art SISR methods contain millions of parameters and layers, which limits their practical applications. In this paper, we propose a hardware (Synaptics Dolphin NPU) limitation aware, extremely lightweight quantization robust real-time super resolution network (**XLSR**). The proposed model's building block is inspired from root modules introduced in [15] for Image classification. We successfully applied root modules to SISR problem, further more to make the model uint8 quantization robust we used Clipped ReLU at the last layer of the network and achieved great balance between reconstruction quality and runtime. Furthermore, although the proposed network contains 30x fewer parameters than VDSR [16] its performance surpasses it on Div2K validation set. The network proved itself by winning Mobile AI 2021 Real-Time Single Image Super Resolution Challenge.*

1. Introduction

Super Resolution is a classical computer vision problem and it has been studied over decades. The aim of the problem is obtaining the high resolution image from either single or multiple low resolution images. In both settings the problem is ill-posed. Earlier in the literature the problem is approached with traditional methods later with the advancement and success of deep learning, the problem is approached with deep-learning.

Deep learning first applied to SISR problem by Dong et al. in [5] and achieved state-of-the-art results compared to traditional methods with only 3 layers. Researchers realized that upscaling the input image on the later stages of the network reduces computational resources and proposed FSRCNN [6] further more instead of ReLU, PReLU is utilized as the activation function, while FSRCNN used deconvolution layer as the upscaling module. Shi et al. [24] proposed another widely used upscaling layer so called Depth2Space for real-time SISR problem. Later, VDSR [16] emerged and increased the number of layers and number of parame-

(a) Original: Div2K 0890



(b) **XLSR (ours)** (c) ESPCN (d) FSRCNN

Figure 1: Effect of quantization of existing methods and our proposed method. Note the shift in colors and staircase gradient on the sky of quantized ESPCN and FSRCNN outputs

Method	Number of Parameters
EDSR [20]	43M
WDSR [28]	75M
VDSR* [16]	668K
IMDN [11]	500K
FSRCNN* [6]	25K
ESPCN* [24]	31K
XLSR (Ours)	22K

Table 1: Example set of number of parameters from high performing deep learning networks.
(*) For a fair comparison of parameters model is assumed to accept RGB input and scaling is x3

ters and showed that with the increased number of parameters the reconstruction quality can be improved. Increasing and complicating the network architecture trend continued later on with EDSR [20] and WDSR [28] later on attention mechanisms applied to SISR in [22, 11] Indeed these methods achieved superior reconstruction performance on standard datasets.

While previously mentioned methods focus on reconstruction quality, there are also other methods based on Generative Adversarial Networks (GAN). GAN based methods [18, 26] focuses on perceptual quality.

However, none of these methods and many more in the literature can be directly applied and run in mobile devices either because of extremely large number of parameters and/or severely affected performances due to uint8 quantization, further more some specific hardware limitations may also limit their applicability.

To solve the real-time image super resolution problem with mobile device deployment requirement, we proposed an extremely lightweight super resolution (**XLSR**) network by investigating the limiting factors of the existing models to run in mobile devices and modifying successfully applied mobile network building blocks for different problems in the literature (See Figure 2). Our method designed with reconstruction quality focus since Mobile AI 2021 Real-time image super resolution challenge scoring formula was based on PSNR as follows;

$$Score(PSNR, runtime) = \frac{2^{2.PSNR}}{C.runtime} \qquad (1)$$

Where C is a constant

Building an extremely lightweight network with low number of parameters was not enough by itself since the model for the challenge needs to be fully uint8 quantized. To make the model quantization robust and keep it still running fast on the deployment platform, instead of "Linear" activation function, which is common at the very last layer of SISR deep learning models, we used "Clipped ReLU" and carefully tuned training methodology. Note that this is required since added non-linearity at the last layer, although helps with quantization, unfortunately makes the model optimization harder due to extra flat optimization surface. It is shown that the model trained with this mentality is very robust to quantization and only $\sim$0.3dB PSNR drop is observed on the quantized model with standard tensorflow post training quantization compared to float16/32 model's PSNR on Div2K validation dataset.

2. Related Works

Successful deployment of deep learning models to mobile devices opens broader application areas to these models and increases their usability along with academic contribution. These motivations lead to advancements on both AI specific hardware and on the mobile friendly models. Hardware focusing on deep learning deployment started with the efforts of Qualcomm and Arm later on continued with specialized AI silicon from many different vendors [14]. On the other hand, many researchers focusing on mobile deployment, come up with many different ideas for mobile friendly models. The examples of these ideas are; for Object detection SqueezeNet [12] which achieves AlexNet [17] performance on ImageNet with 50x lesser number of parameters. MobileNetV1 [9] uses depthwise separable convolutions and achieves better performance than SqueezeNet. ShuffleNet [31] further increases ImageNet performance with lighter requirements by utilizing group convolutions and channel shuffling operator. Similar to ShuffleNet, DeepRoots proposes the usage of 1x1 convolutions instead of channel shuffling. For face verification, MobileFaceNets [3] uses depthwise separable convolutions and bottleneck layers to build an application specific lightweight network. For image super resolution IMDN [11] uses channel splitting to build a lighter network. Different from previous approaches, NASNet [33] searches for the optimal architecture and surpasses many state-of-the-art methods.

Apart from focusing on network itself, the literature also focuses on quantization [32, 10, 4, 19] since in many cases quantization is not an option but a hardware declared must. On the other hand, it has been shown by Hinton et al. [8] that knowledge distillation can help a simple model to achieve/surpass a complex model's performance. Furthermore, Mishra et al. [21] combined knowledge distilling with quantization. Another methodology using a pretrained complex model while building a lighter weight model is; channel sparsification and pruning, with this methodology a complex model can be slimmed by removing unnecessary channels from the filters [7]

Although the most of the aforementioned methods are

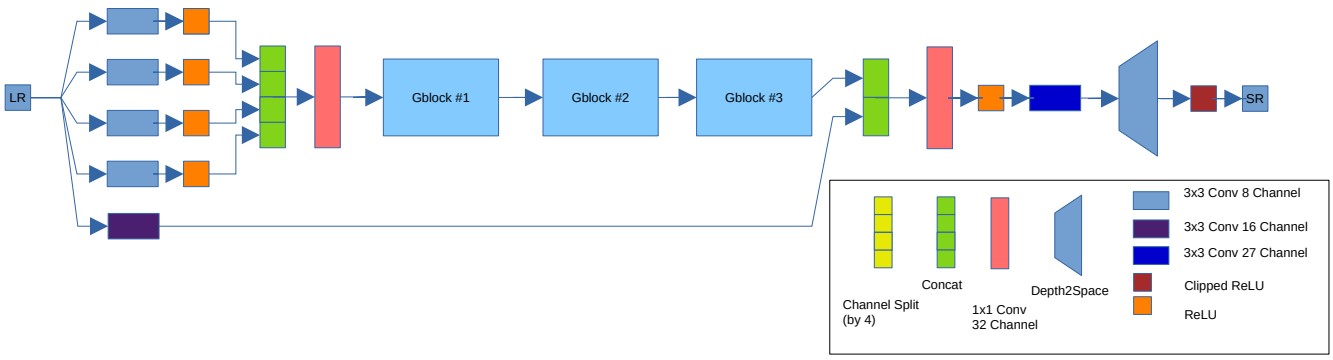


Figure 2: Our proposed network

designed for different areas other than SISR, they can be still very useful while building a real-time performant SISR model. Many of these ideas are succesfully applied to the SISR problem [30, 29] The methodologies to run/design a performant deep learning method for mobile devices can be summarized as follows;

- Hand-Designed Architectures
- Efficient Building Block Design
- Network Pruning / Sparsification
- Network Quantization
- Network Architecture Search (NAS)
- Knowledge Distillation

In this paper, we followed the first and second methodologies while creating our submission into Mobile AI 2021 Real-Time Single Image Super-Resolution Challenge [13]. The reason and motivation for this approach was; there were many different hardware limitations of the deployment platform (Synaptics Dolphin NPU) that needs to be taken into account in the challenge which are harder to incorporate into a common method. Furthermore, the challenge required full uint8 quantization of the model (not allowing partial quantization such as leaving the first and the last layers floating, which are known to be severely affecting the accuracy if quantized [4]). The full uint8 quantization requirement adds extra complexity to the problem since, SISR problem is severely affected by the quantization operation if the model at the hand is not designed/trained/quantized properly.

The deployment hardware limiting factors can be summarized as

- Elementwise operations such as (Addition, Subtraction) is not optimized
- Reshaping and transpose operations are not optimized and extremely slow
- Per-channel quantization is not supported
- Multiple and especially long skip connections require a lot of data swaps with slow CPU DRAM

3. Proposed Method

In this section, we describe the details of the proposed network and the motivation behind the design ideas while connecting these with literature and hardware limitations. As mentioned before, we designed our proposed architecture by hand and adopted an efficient building block for SISR problem inspired from [31, 15, 27].

3.1. Building Block Selection

Group convolutions are first used in AlexNet although the GPU hardware limitation forced such methodology. It has been shown that when used wisely, group convolutions can increase accuracy while decreasing computational costs. Because of these properties, they are often used in mobile focused networks. They are used in ResNeXt along with skip connections, in ShuffleNet with cascaded channel shuffling and in DeepRoots with cascaded 1x1 convolutions.

From the point of the view of the SISR problem challenge and hardware limitations, using channel shuffling is infeasible since reshape and transpose operations are not optimized in the deployment hardware which eliminates ShuffleNet Block. Skip connections and residual in residual type of structures help with model converge and allow deeper architectures and are utilized in many state-of-the-art networks. However, multiple parallel skip connections are slow and elementwise addition is not optimized and hence ResNeXt block is not very well optimized for the deployment hardware.

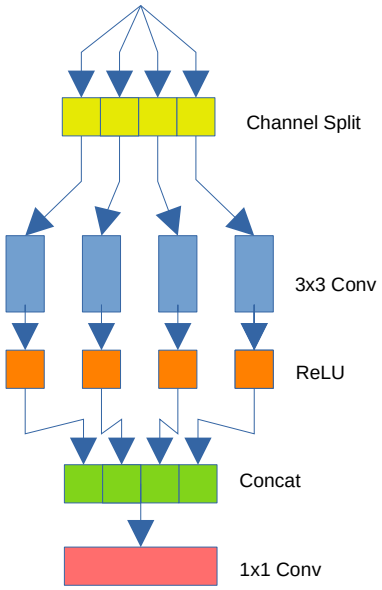


Figure 3: Building block used in the proposed network

On the other hand, when channel shuffling operator is relaxed to 1x1 convolution (to still allow interchannel communication) or skip connection in ResNext block is removed, we arrive at the building block used in our network (See Figure 3). Note that instead of group convolutions, depthwise convolutions can also be a candidate for building block. However, as noted by Sheng et al. in [23] depthwise convolutions can cause large quantization error when used without special precaution, this is what we have also observed empirically in our experiments.

An important aspect we have also taken into account while designing our proposed model was elementwise operations are not optimized in the deployment hardware. Thus, we avoided all addition and scaling operations and used concatenation operation when necessary and input data scaling and normalization were not used.

3.2. Quantization Friendly Architecture

While deploying deep learning models to mobile devices, it is very common to use quantization especially 8-bit quantization is very important since it is naturally supported by many mobile hardware as with the deployment hardware of Mobile AI 2021 Real-time single image super resolution challenge. However, applying uint8 quantization scheme to any well performing (in float32 or float16) super resolution model and hoping to arrive at a well performing integer quantized model is very naive and usually not working well in practice (See Figure 4). To get over this problem and still have a well performing model, the model should be modified and made quantization friendly. On the other hand, this modification should be subtle and still be quickly executable, otherwise it would contradict with real-time inference requirement.

To make a model quantization friendly, we should first focus on the reasons behind why quantization adversely affects the accuracy. Linear output activation function is very common among super resolution models and it helps with the model optimization. Although, not strictly enforced when the model converges we are pretty sure that the output will be bounded in between 0-255 (or 0-1.0). However, if such a model is quantized, the quantized output tend to be dull and accuracy can drop about 5-7dB compared to float16/32 model. The reason we believe is the following; during the earlier steps of training, the output is not guaranteed to be in between 0-1.0 and intermediate activations can also be unbounded or might contain outliers. Later on, if we continue with training the training data enforces boundedness in the model output indirectly. However, the intermediate activations of the model are not acted upon and the model may converge to a point near where its intermediate activations are unbounded, since model usually visits these points in the early stages of the training. These allowed intermediate unbounded activations create outliers (very large a few numbers). The outliers in intermediate layer activations, when uint8 quantized leads to some important information carrying, comparably low amplitude values to be zeroed out. Hence effective signal energy reaching to the last layer drops, which results in dull colors and drastic PSNR drops.

With this motivation we believe if the intermediate layer activations are forced not to visit these outlier creating points from the beginning of the training, the model would be more quantization friendly. This is exactly what happens when the model is trained with Clipped ReLU at the last layer. A model trained with this mentality results in only $\sim$0.2-0.5dB loss with respect to its floating counterpart. This is better seen in Table 2 where we trained our proposed model without Clipped ReLU and quantized the resulting model and noted Div2K [1] validation accuracy.

Activation	PSNR (fp32)	PSNR (uint8)	Drop
Linear	30.11	24.72	5.39
Clipped ReLU	**30.10**	**29.82**	**0.28**

Table 2: Effect of Clipped ReLU and Div2K Validation Set PSNR Results with floating and uint8 quantized models

Note that Clipped ReLU is

$$Clipped\_ReLU(x) = max(0, min(x, 1)) \quad (2)$$

so it is very cheap to calculate. Thus the computational burden added to the model is minimal which is desired.

Furthermore, although placing a single Clipped ReLU was enough to regularize the intermediate activations of our proposed network, when a model gets deeper, regularization effect might vanish for deeper layers, to overcome this issue we suggest to change a few of the ReLU's with Clipped ReLU's. We believe that clipping only a few ReLU's is enough to regularize the intermediate activations though we have not experimented on this idea for this paper. Also note that for intermediate Clipped ReLU's clipping value does not need to be 1 and it should either be experimentally found or it should be included in the optimization as done in [19].

One drawback of using Clipped ReLU at the output layer is unfortunately the model is harder to optimize since it creates a lot of flat regions with minimal or no derivative direction. Because of this reason, to converge to a better model, a few training tricks needs to be employed which are explained in training details.

4. Experiments

4.1. Datasets

We used the supplied Div2K Dataset [1] for the challenge and no extra data is used. The dataset consist of 800 high quality training images and 100 validation images along with 100 test images. During the challenge the test images were not released. Due to this, while reporting our results we only used validation set results (which are not used either for training or validation). For a fair comparison, on standard benchmark datasets (Set5, Set14, BSD100, Manga109, Urban100) with the existing work, we used our floating point model and used Y channel of our output.

4.2. Training Details

For training, we splitted 800 training images into two sets, we used the first 792 of training images for training while keeping last 8 images for validation. We cropped random 32x32 LR images and for data augmentation we used geometric transformations (8 transformations - original, rotations, flips) with equal probability. Furthermore, to give more robustness to illumination changes we randomly scaled the intensity of images with 1, 0.7 and 0.5 randomly. As the loss function, we used Charbonnier loss [2] with $\epsilon = 0.1$ as defined in (3) since it is the smoother version of L1 and we empirically found out that it works better with Clipped ReLU

$$Charbonnier(x) = \sqrt{x^2 + \epsilon^2} \tag{3}$$

As mentioned above, the critical part of the model development was to include Clipped ReLU activation function. This is however on the other hand results in a harder problem. So to still be able to converge to a performant model following tricks were utilized;

Method	PSNR	SSIM	Runtime	Score
XLSR (ours)	29.58	0.86	44.85ms	51.02
2nd Method	29.41	0.8537	38.32ms	47.18
3rd Method	29.52	0.8607	62.25ms	33.82
4th Method	28.82	0.8428	76.61ms	10.41
5th Method	28.92	0.8486	718.32ms	1.28

Table 3: Mobile 2021 Real-Time Single Image Super Resolution Results and Scores on Div2K Test Dataset (not released)

- We used a triangular cyclic learning rate scheduling strategy [25], which starts with a very low value (5e-5) for the first epoch and quickly increases to the top value(25e-4) in 50 epochs and slowly decreases to a low value till 5000 epochs (1e-4).
- Trained the model with mini batch size 16 for 5000 epochs with each epoch containing 100 mini batches. At the end of each epoch, we calculated the validation set PSNR (last 8 images of training set) and saved the best model for quantization
- Used Adam optimizer with default beta1=0.9 and beta2=0.999 and epsilon=1e-8 values
- Initialized Conv2D layers with He-Normal with 0.1 Variance Scaling [20] The motivation behind this was to initialize kernels to closer to zero and avoid large numbers which might in turn create outliers in activation

For quantization, we used standard Tensorflow Post-Quantization strategy and trained the model on NVIDIA RTX 2080 Super. It takes about 2 hours to train our proposed model in that hardware.

Our quantized model achieves the following runtimes in Samsung Galaxy A21s using AI-Benchmark Tool [1] [14];

- CPU Runtime 1130ms
- NNAPI Runtime 1150ms
- GPU Delegate Runtime 620ms

XLSR also achieves an exceptional target platform runtime of 45ms. Table 3 shows the results from Mobile AI 2021 Real-Time Super Resolution Challenge [13].

PSNR results of our proposed method, compared with some of the work in the literature on the standard benchmark dataset and Div2K validation set can be seen in Table 4. Note that for a fair and consistent comparison with the literature, the result presented here are from our float32 model. Comparison of our quantized model with quantized FSRCNN and ESPCN can be found in Table 5.

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

Dataset	Scale	Bicubic	FSRCNN	ESPCN	VDSR	IMDN	EDSR	**XLSR (ours)**
Set5	x3	30.41	33.16	33.13	33.66	34.36	34.65	33.42
Set14	x3	27.55	29.43	29.42	29.77	30.32	30.52	29.73
B100	x3	27.22	28.60	28.50	28.82	29.09	29.25	28.55
Urban100	x3	24.47	26.48	26.41	27.14	28.17	28.80	26.71
Manga109	x3	26.99	30.98	30.79	32.01	33.61	-	31.63
Div2K(Val)	x3	28.22	-	-	30.09	-	31.26	**30.10**

Table 4: Comparison of our proposed method with public benchmark scores of other methods. Note that for fair and consistent comparison with the literature, we used our floating point model and and converted RGB output of our method to Luminance (Y) channel only while calculating PSNR for all dataset except Div2K

(a) Original: Div2K 0833

(b) HR

(d) **XLSR (ours)**

(c) ESPCN

(e) FSRCNN

(f) Original: Div2K 0823

(g) HR

(i) **XLSR (ours)**

(h) ESPCN

(j) FSRCNN

Figure 4: Example Images from Div2K Dataset. Note false colors and staircase gradient effect on quantized ESPCN and FSRCNN output

Model	FSRCNN* (dB)	ESPCN* (dB)	XLSR (dB)
float32	29.67	29.54	**30.10**
uint8	21.95	23.93	**29.82**
drop	7.72	5.61	**0.28**

Table 5: Effect of quantization on the existing methods and our method on Div2K validation (x3 scale) set. (*) Our implementation and training of the models to convert them to RGB input and RGB output

5. Conclusion

In conclusion, we proposed a real-time single image super resolution method driven by the hardware constraints of the Mobile AI 2021 challenge, although it is driven by the target hardware, the resulting model is very efficient in terms of runtime and model parameters. We believe it can run in many mobile hardware with high performance. The proposed model easily surpasses many reported PSNR results of famous FSRCNN and ESPCN models and it even reaches VDSR in most of the public datasets and surpasses its performance on Div2K validation set though it has 30x fewer parameters. Furthermore, the proposed architecture is very robust to uint8 to quantization and only 0.28dB PSNR drop is experienced when compared with float16/32 model. This property of the model makes it a really good candidate for many mobile devices.

References

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *IEEE Conf. Comput. Vis. Pattern Recog. Worksh.*, July 2017.

[2] P. Charbonnier, L. Blanc-Feraud, G. Aubert, and M. Barlaud. Two deterministic half-quadratic regularization algorithms for computed imaging. *Proceedings of 1st International Conference on Image Processing*, 2:168–172 vol.2, 1994.

[3] Sheng Chen, Yang Liu, Xiang Gao, and Zhen Han. Mobilefacenets: Efficient cnns for accurate real-time face verification on mobile devices. *Biometric Recognition - 13th Chinese Conference, CCBR*, pages 428–438, 2018.

[4] Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I. Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. Pact: Parameterized clipping activation for quantized neural networks. *Int. Conf. Learn. Represent.*, 2018.

[5] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. *Eur. Conf. Comput. Vis.*, 8692:184–199, 2014.

[6] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. *Eur. Conf. Comput. Vis.*, 9906:391–407, 2016.

[7] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Int. Conf. Comput. Vis.*, pages 1398–1406, 2017.

[8] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *Adv. Neural Inform. Process. Syst.*, 2015.

[9] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017.

[10] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. *J. Mach. Learn. Res.*, 18:187:1–187:30, 2017.

[11] Zheng Hui, Xinbo Gao, Yunchu Yang, and Xiumei Wang. Lightweight image super-resolution with information multi-distillation network. *ACM Int. Conf. Multimedia*, 2019.

[12] Forrest N. Iandola, Matthew W. Moskewicz, Khalid Ashraf, Song Han, William J. Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and 1mb model size. *Int. Conf. Learn. Represent.*, 2017.

[13] 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.

[14] 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. pages 3617–3635, 2019.

[15] Yani Ioannou, Duncan P. Robertson, Roberto Cipolla, and Antonio Criminisia. Improving cnn efficiency with hierarchical filter groups. *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5977–5986, 2017.

[16] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1646–1654, 2016.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. 2012.

[18] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew P. Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 105–114, 2019.

[19] Huixia Li, Chenqian Yan, Shaohui Lin, Xiawu Zheng, Yuchao Li, Baochang Zhang, Fan Yang, and Rongrong Ji. Pams: Quantized super-resolution via parameterized max scale. *Eur. Conf. Comput. Vis.*, 2020.

[20] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. *IEEE Conf. Comput. Vis. Pattern Recog. Worksh.*, pages 1132–1140, 2017.

[21] Asit K. Mishra and Debbie Marr. Apprentice: Using knowledge distillation techniques to improve low-precision network accuracy. In *Int. Conf. Learn. Represent.*, 2018.

[22] Ben Niu, Weilei Wen, Wenqi Ren, Xiangde Zhang, Lianping Yang, Shuzhen Wang, Kaihao Zhang, Xiaochun Cao, and Haifeng Shen. Single image super-resolution via a holistic attention network. *Int. Conf. Comput. Vis.*, 2020.

[23] Tao Sheng, Chen Feng, Shaojie Zhuo, Xiaopeng Zhang, Liang Shen, and Mickey Aleksic. A quantization-friendly separable convolution for mobilenets. *CoRR*, abs/1803.08607, 2018.

[24] Wenzhe Shi, Jose Caballero, Ferenc Huszar, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1874–1883, 2016.

[25] L. N. Smith. Cyclical learning rates for training neural networks. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 464–472, 2017.

[26] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. pages 63–79, 2018.

[27] Saining Xie, Ross B. Girshick, Piotr Dollar, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5987–5995, 2017.

[28] Jiahui Yu, Yuchen Fan, Jianchao Yang, Ning Xu, Zhaowen Wang, Xinchao Wang, and Thomas S. Huang. Wide activation for efficient and accurate image super-resolution. *IEEE Conf. Comput. Vis. Pattern Recog. Worksh.*, 2018.

[29] Kai Zhang, Martin Danelljan, Yawei Li, Radu Timofte, Jie Liu, Jie Tang, Gangshan Wu, Yu Zhu, Xiangyu He, Wenjie Xu, Chenghua Li, Cong Leng, Jian Cheng, Guangyang Wu, Wenyi Wang, Xiaohong Liu, Hengyuan Zhao, Xiangtao Kong, Jingwen He, Yu Qiao, Chao Dong, Xiaotong Luo, Liang Chen, Jiangtao Zhang, Maitreya Suin, Kuldeep Purohit, A. N. Rajagopalan, Xiaochuan Li, Zhiqiang Lang, Jiangtao Nie, Wei Wei, Lei Zhang, Abdul Muqeet, Jiwon Hwang, Subin Yang, JungHeum Kang, Sung-Ho Bae, Yongwoo Kim, Liang Chen, Jiangtao Zhang, Xiaotong Luo, Yanyun Qu, Geun-Woo Jeon, Jun-Ho Choi, Jun-Hyuk Kim, Jong-Seok Lee, Steven Marty, Eric Marty, Dongliang Xiong, Siang Chen, Lin Zha, Jiande Jiang, Xinbo Gao, Wen Lu, Haicheng Wang, Vineeth Bhaskara, Alex Levinshtein, Stavros Tsogkas, Allan Jepson, Xiangzhen Kong, Tongtong Zhao, Shanshan Zhao, Hrishikesh P S, Densen Puthussery, Jiji C V au2, Nan Nan, Shuai Liu, Jie Cai, Zibo Meng, Jiaming Ding, Chiu Man Ho, Xuehui Wang, Qiong Yan, Yuzhi Zhao, Long Chen, Jiangtao Zhang, Xiaotong Luo, Liang Chen, Yanyun Qu, Long Sun, Wenhao Wang, Zhenbing Liu, Rushi Lan, Rao Muhammad Umer, and Christian Micheloni. Aim 2020 challenge on efficient super-resolution: Methods and results. pages –, 2020.

[30] Kai Zhang, Shuhang Gu, Radu Timofte, Zheng Hui, Xiumei Wang, Xinbo Gao, Dongliang Xiong, Shuai Liu, Ruipeng Gang, Nan Nan, Chenghua Li, Xueyi Zou, Ning Kang, Zhan Wang, Hang Xu, Chaofeng Wang, Zheng Li, Linlin Wang, Jun Shi, Wenyu Sun, Zhiqiang Lang, Jiangtao Nie, Wei Wei, Lei Zhang, Yazhe Niu, Peijin Zhuo, Xiangzhen Kong, Long Sun, and Wenhao Wang. Aim 2019 challenge on constrained super-resolution: Methods and results. pages –, 2019.

[31] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018.

[32] Shuchang Zhou, Zekun Ni, Xinyu Zhou, He Wen, Yuxin Wu, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *CoRR*, abs/1606.06160, 2016.

[33] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. Learning transferable architectures for scalable image recognition. *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 8697–8710, 2018.