Enhanced Deep Residual Networks for Single Image Super-Resolution

CVPR 2017

- It uses a modified version of ResNet architecture. It removes batch normalization layers from residual blocks.
- Each residual block consist of conv1-Relu-conv2 and skip connnections.

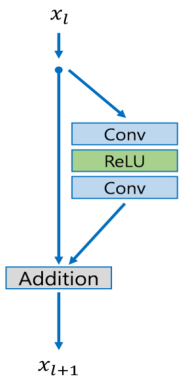

Fig.1: residual block used in EDSR

- To increase training stability, residual block scaling added as shwon in Fig.2.

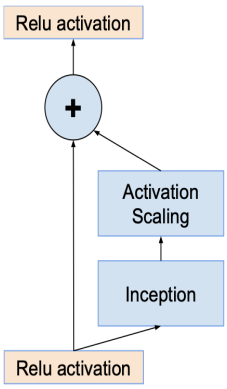

Figure 20. The general schema for scaling combined Inception-resnet moduels. We expect that the same idea is useful in the general resnet case, where instead of the Inception block an arbitrary subnetwork is used. The scaling block just scales the last linear activations by a suitable constant, typically around 0.1.

Fig.2: Residual block scaling

- They want to increase network layer(depth) instead of increasing number of feature channel(width) because of increasing width instead of depth increases model parameters more.
- **Important:** There is no RELU layer outside the residual blocks.
- In final model: **depth:** 32 layers, **width:** 256 filters, **scale factor:** 0.1
- Training speed can be accelerating for x3 and x4 upscaling factors using pretrained x2 network parameters at initialization. (like transfer learning - fine tuning)
- For single scale model which is called as EDSR, for each scale factor, network is trained separately as shown in Fig.3. Also, Fig.3 shows general architecture of EDSR.

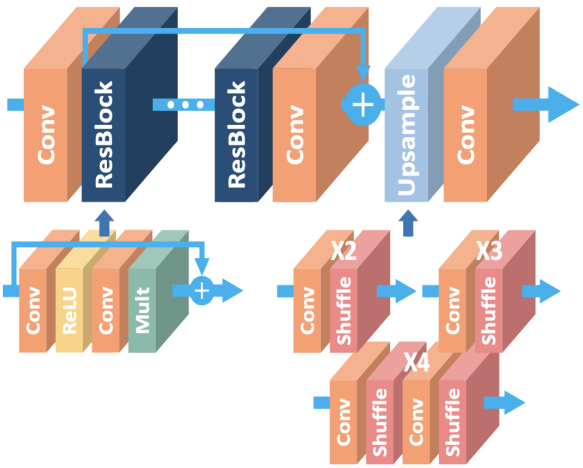

Figure 3: The architecture of the proposed single-scale SR network (EDSR).

Fig.3: Single Scale Model(EDSR)

- There is also multi-scale model which is called **MDSR**. Fig.4 shows the architecture of the MDSR and models(MDSR/EDSR) are compared.

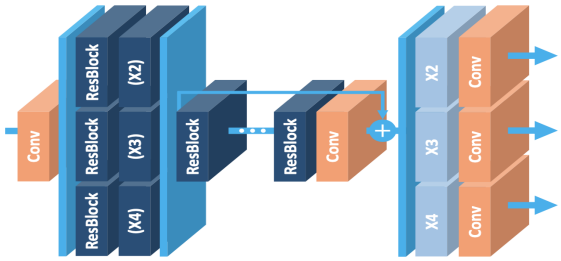

Figure 5: The architecture of the proposed multi-scale SR network (MDSR).

Options	SRResNet [14] (reproduced)	Baseline (Single / Multi)	EDSR	MDSR
# Residual blocks	16	16	32	80
# Filters	64	64	256	64
# Parameters	1.5M	1.5M / 3.2M	43M	8.0M
Residual scaling	-	-	0.1	-
Use BN	Yes	No	No	No
Loss function	L2	L1	L1	L1

Fig.4: Multi Scale Model(MDSR)

-**Training:** DIV2K dataset is used

- Input: RGB 48x48 patches
- Augment training data with rotations and random flips
- Preprocess all images by subtraction mean RGB value
- ADAM optimizer is used($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$).
- Batch size: 16
- Learning rate: 10^{-4} initially and halved at every $2 * 10^5$ batch updates.
- For EDSR model, first x2 model is trained and x3 and x4 model uses pretrained weights from x2 model for initialization.
- For MDSR, batch is randomly selected from x2, x3 and x4. Then, only selected scale parameters are enabled and activated.
- L1 loss is used instead of L2 loss because of stability issues.(L1 provides better convergence).

- Results and Project Codes