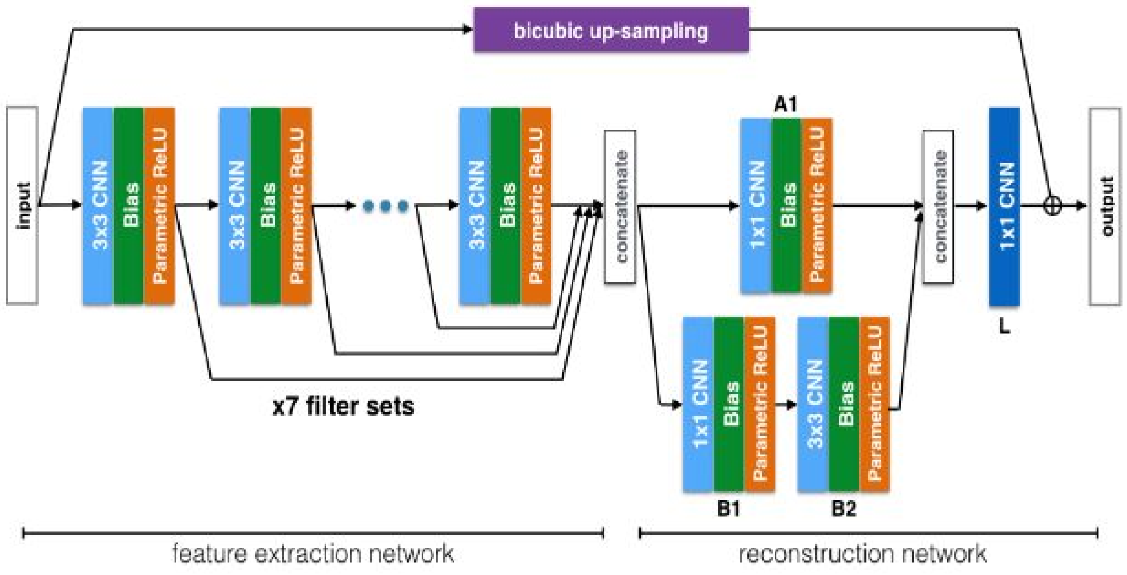
Fast and Accurate Image Super Resolution by Deep CNN with Skip Connection and Network in Network

Project INF573
Prepared by Dmitry Gaynullin

Project Description

The purpose of our project is the transformation of low resolution images to the high resolution ones (Single Image Super-Resolution (SISR)). Previously, Single Image Super-Resolution was used in specialized areas such as medical imaging and video surveillance for security purposes, but this type of networks could be very useful for transforming videos and websites on the fly, as a screen resolution is constantly increasing and an old content needs to be adapted to new screens. The solution [1] should be efficient, so, in order to do this, we’ve chosen a model with convolutional neural networks, because of their best achieved reconstruction performance. Traditionally, the most performant reconstruction networks require a lot of processing power, therefore, at the same time, another basic criteria for this solution are the relative lightness and less dependence on computing power so that this network could be suitable for the portable devices. Thus, this model achieves 10 times lower computational cost with the state-of-the-art performance than the model based on classic Deep CNN with Residual Net, Skip Connection and Network in Network. Skip connection layers and CNNs used for image feature extraction locally and globally, and so called Network in Network layers (1*1 CNNs used in parallel) for image reconstruction. In order to obtain faster calculation rates with less information loss, each convolution network has been optimized (reducing the number of layers and filters).

Architecture



This network consists of two main parts - the feature extraction and the reconstruction networks. Firstly, we calculate a bicubic upsampling of an input image then we enchain blocks of CNN and ReLu activation and, at the same time, extract the information from each layer's output to get more precise results. And after the chain of these CNN and ReLu blocks, we concatenate the results from each of them all together and pass the result further to the reconstruction network.

In the reconstruction part, we split the calculation path into two tracks: A1 and B1 -> B2 to reconstruct even better. 1x1 CNN layer reduces the dimensions of the previous layer and also adds more nonlinearity to enhance the potential representation of the network. And after concatenation of the results of these two parts, 1 x 1 CNN with four filters are applied for an image reconstruction. After that, we sum up the reconstructed image and a bicubic upsampling of the original one. Finally, we obtain a high resolution picture.

Training details

For training of the model, 291 images from two different datasets have been used (91 images from Yang et al. [2] and 200 images from Berkeley

Segmentation dataset [3]). The data has been augmented by flipping these images horizontally, vertically and both horizontally and vertically at the same time. For validation of the model, another set of images has been used.

The loss function that was used in this article is a mean squared error between the predicted image and the ground truth.

Code and results

First part of the code is about the data preparation. Firstly, we need to align all the images, then have to perform a data augmentation by turning and flipping of all the images and, after that, we crop all the photos by the patches of size 32 x 32 x 1.

Once it's finished we need to get a "good" and "bad" resolution images. For this we use bicubic upsampling and downsampling.

Then we implement the network, loss creation and optimizer initialization. An optimizer needs to be initialized with a very low learning rate to prevent gradient explosion.

After implementation of all necessary parts of this article such as data preparation, creation of the network, losses, optimizer, we have not succeeded to train the network. Even if all the parts work well separately during the training we've got losses with *NaN* values. Different hypothesis were tested to solve this error: different optimizers (rms prop, adam) with very low learning rates, check for inf values in loss, visualization of every layer output, regularisations. After trying all of these approaches, we've succeeded to find the source of this gradient explosion.

References

[1] - https://arxiv.org/pdf/1707.05425.pdf - Fast and Accurate Image Super Resolution by Deep CNN with Skip Connection and Network in Network, Yamanaka et al.
[2] - https://www.columbia.edu/~jw2966/papers/YWHM10-TIP.pdf - Image Super-Resolution via Sparse Representation, Yang et al.
[3] - https://ieeexplore.ieee.org/document/937655 - A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics, Martin et al.