A Hybrid Approach Between Adversarial Generative Networks and Actor-Critic Policy Gradient for Low Rate High-Resolution Image Compression

Nicoló Savioli
Imperial College London, UK
nsavioli@ic.ac.uk

Abstract

Image compression is an essential approach for decreasing the size in bytes of the image without deteriorating the quality of it. Typically, classic algorithms are used but recently deep-learning has been successfully applied. In this work, is presented a deep super-resolution work-flow for image compression that maps low-resolution JPEG image to the high-resolution. The pipeline consists of two components: first, an encoder-decoder neural network learns how to transform the downsampling JPEG images to high resolution. Second, a combination between Generative Adversarial Networks (GANs) and reinforcement learning Actor-Critic (A3C) loss pushes the encoder-decoder to indirectly maximize High Peak Signal-to-Noise Ratio (PSNR). Although PSNR is a fully differentiable metric, this work opens the doors to new solutions for maximizing non-differential metrics through an end-to-end approach between encoder-decoder networks and reinforcement learning policy gradient methods.

1. Introduction

Image compression with deep learning systems is an active area of research that recently has becomes very compelling respect to the modern natural images codecs as JPEG2000, [1], BPG [2] WebP currently developed by Google® [3]. The new deep learning methods are based on an auto-encoder architecture where the features maps, generate from a Convolutional Neural Networks (CNN) encoder, are passed through a quantizer to create a binary representation of them, and subsequently given in input to a CNN decoder for the final reconstruction. In this view, several encoders and decoders models have been suggested as a ResNet [4] style network with the parametric rectified linear units (PReLU) [5], generative approach build on GANs [6] or with a innovative hybrid networks made with Gated Recurrent Units (GRUs) and ResNet [7]. In contrast, this paper proposes a super-resolution approach, build on a modifying version of SRGAN [8], where downsampling JPEG images are converted at High Resolution (HR) images. Hence, in order to improve the final PSNR results, a Reinforcement Learning (RL) approach is used to indirectly maximize the PSNR function with an A3C policy [9] end-to-end joined with SRGAN. The main contributions of this works are: (i) Propose a compression pipeline based on JPEG image downsampling combined with a super-resolution deep network. (ii) Suggest a new way for maximizing not differentiable metrics through RL. However, even if the PSNR metric is a fully differentiable function the proposed method could be used in future applications for non-euclidean distance such as in the Dynamic Time Warping (DTW) algorithms [10].

2. Methods

In this section is given a more formal description of the suggested system which includes: the network architecture and the losses used for training.

2.1. Network Architecture

The architecture consists of three main blocks: encoder, decoder and a discriminator (Figure 1). Where \( I^{LR} \) is the low-resolution (LR) input image (i.e compressed with a JPEG encoder) of size \( rW \times rH \times C \) (i.e with C color channels and W, H the image width and height); where a bicubic downsampling operation with factor \( r \) is applied. While the output is an HR image defined as \( I^{HR} \).

2.1.1 Encoder

The encoder is basically a ResNet [4], where the first convolution block has a kernel size of \( 9 \times 9 \) and 64 feature maps with a \textit{ParametricReLU} activation function. Then, five Residual Blocks (RB) are stacked together. Each of those RB consists of two convolution layers with kernel size \( 3 \times 3 \) and 64 feature maps followed by Batch-Normalisation (BN) and \textit{ParametricReLU}. After that, a final convolution block of \( 3 \times 3 \) and 64 feature maps are repeated. However, the encoder is also joint with a fully connected layer.
and, at each \(i\) training iterations, produces an action prediction of the actual PSNR together with a value function \(V^\pi(I_{HR}^i)\) (i.e., explained in 2.2.2 section).

### 2.2. Loss function

The accurate definition of the loss function is crucial for the performance of the \(H(\cdot)_\theta\) generator. Here, we divide the paragraph into three losses: the SRGAN loss, the RL loss, and the proposed loss.

#### 2.2.1 SRGAN loss

The SRGAN loss is determined as a combination of three other separate losses: MSE loss, VGG loss, and GANs loss. Where the MSE loss is defined as:

\[
l_{HR}^{MSE} = \frac{1}{W H} \sum_{x=1}^{W} \sum_{y=1}^{H} (I_{HR}^{x,y} - H_\theta(I_{LR}^{x,y}))^2 \tag{2}
\]

It represents the most utilized loss in super-resolution methods but remarkably sensitive to high-frequency peak with smooth textures [8]. For this reason, is used a VGG loss [8] based on the ReLU activation function of a 19-layer VGG (defined here as \(\Omega(\cdot)\)) network:

\[
l_{HR}^{VGG} = \frac{1}{W H} \sum_{x=1}^{W} \sum_{y=1}^{H} (\Omega(I_{HR}^{x,y}) - \Omega(H_\theta(I_{LR}^{x,y})))^2 \tag{3}
\]

Where \(W\) and \(H\) are the dimension of \(I_{HR}\) image in the MSE loss. Whilst, for the VGG loss, they are the \(\Omega(\cdot)\) output feature maps dimension. While the GANs loss is previously defined in the equation 1. Finally, the total SRGAN loss is determined as:

\[
l_{SRGAN}^{HR} = l_{MSE}^{HR} + 1e - 3 \times l_{GAN}^{HR} + 6e - 3 \times l_{VGG}^{HR} \tag{4}
\]
2.2.2 RL loss

The aim of RL loss is to indirectly maximize the PSNR through an actor-critic approach [9].

Given \( Q^\pi(I^{LR}, PSNR_{pred}) \) a map between the low-resolution input \( I^{LR} \) and the current PSNR value prediction \( PSNR_{pred} \) (see fig. 1). Then, at each \( i \) training iteration, is calculated the reward value as a threshold between the previous PSNR at iteration \( i-1 \) and that one to iteration \( i \) as follows:

\[
r(i) = \begin{cases} 1, & \text{if } PSNR_i > PSNR_{i-1} \\ 0, & \text{otherwise} \end{cases}
\]

where the \( PSNR(i) \) function is defined as:

\[
PSNR = 20 \cdot \log_{10}(MAX_I) - 10 \cdot \log_{10}\left( \frac{1}{mn} \sum_{l=0}^{m-1} \sum_{j=0}^{n-1} (|I^{HR}(l,j) - I^{gt}_l(l,j)|)^2 \right)
\]

The \( MAX_I \) is the maximum pixel value of the HR image, \( I^{HR} \) is the output encoder HR image, while \( I^{gt}_l \) is the corresponding HR ground truth for each pixel \((l,j)\) at \( m \times n \) HR size.

The reward (eq. 5), actually depends on the \( PSNR_{pred} \) action taken by the policy for two main reasons: (i) during the training process the \( PSNR_{pred} \) becomes an optimal estimator of the decoder output \( I^{HR} \) (used in 6). (ii) The latent space between the encoder and the fully connected layer is the same and then share equal policy information. Thus, all the rewards are accumulated every \( k \) training steps through the following return function:

\[
R(i) = \sum_{k=0}^{\infty} \gamma^k r(i+k)
\]

where \( \gamma \in (0,1] \) is a discount factor. It is therefore possible to define the \( Q^\pi(\cdot) \) function as an expectation of \( R(k) \) given the input \( I^{LR} \) and \( PSNR_{pred} \).

\[
Q^\pi(I^{LR}, PSNR_{pred}) = E[R(i)|I^{LR}(i) = I^{LR}, PSNR_{pred}]
\]

To notice, the encoder, together with the fully connected layer, become the policy network \( \pi(PSNR_{pred}|I^{LR}(i); \theta_{\pi}) \). This policy network is parametrized by the standard REINFORCE method on the \( \theta \) encoder parameters with the following gradient direction:

\[
\nabla_{\theta} \log \pi(PSNR_{pred}|I^{LR}(i); \theta) \cdot R(i)
\]

It can be consider an unbiased estimation of \( \nabla_{\theta} E[R(i)] \). Especially, to reduce the variance of this evaluation (and keeping it unbiased) is desirable to subtract, from the return function, a baseline \( V(I^{LR}(i)) \) called value function. The total policy agent gradient is given by:

\[
i^HR_\pi = \log \pi(PSNR_{pred}|I^{LR}(i), \theta) \cdot (R(i) - V(I^{LR}(i)))
\]

The term \( R(i) - V(I^{LR}(i)) \) can be considered an estimation of the advantage to predict \( PSNR_{pred} \) for a given \( I^{LR}(i) \) input. Consequently, a learnable estimation of the value function is used as \( V(I^{LR}(i)) \approx V^\pi(I^{LR}(i)) \). This approach, is further called generative actor-critic [9] because the \( PSNR_{pred} \) prediction is the actor while the baseline \( V^\pi(I^{LR}(i)) \) is its critic. The RL loss is then calculated as:

\[
i^HR_{\text{RL}} = 5e - 3 \cdot \sum (R(i) - V^\pi(I^{LR}(i)))^2 - i^HR_\pi
\]

2.2.3 Proposed loss

The Proposed Loss (PL) combines both SRGAN loss and RL loss. After every \( k \) step (i.e due to the rewards accumulation process at each \( i \) training iterations), the \( i^HR_{\text{RL}} \) is added on \( i^HR_{\text{SRGAN}} \).

\[
i^HR_{\text{PL}} = \begin{cases} i^HR_{\text{SRGAN}} + i^HR_{\text{RL}}, & \text{if } k = i \\ i^HR_{\text{SRGAN}}, & \text{otherwise} \end{cases}
\]

Figure 2: The above figure shows the results for RL-SRGAN compared to SRGAN, LANCZOS, and the original ground truth. As we can see, LANCZOS simply destroys the edges producing artifacts on the global image. Whilst, SRGAN forms noticeable chromatic aberration (i.e transition from orange to yellow color) near the edges. Even though, the RL-SRGAN holds the color uniform with net outline details nearby to the edges; analogous to the original ground truth.

2.3. Experiments and Results

In this section is evaluate the method suggested. The dataset used is the CLIC compression dataset [12] correspondingly divided in the train, valid and test sets. The
A modified version of SRGAN is suggested where an A3C method is joined with GANs. Sadly, the proposed method has strong limitations due to the drastic downsampling of the input JPEG image. This downsampling causes loss of information, difficult to recover from the super-resolution network, which leads to lower results in PSNR and MS-SSIM on the test set (i.e. 20.06 and 0.7503 respectively). Despite, the results (table 1) emphasize slight improvement performances for RL-SRGAN related within SRGAN and a baseline LANCZOS upsampling filter. However, the proposed method processes all test files in a parsimonious way respect to the challenge methods. Indeed, the total dimension of the compression test set is of 5236870 bytes respect to 15748677 bytes of CLIC 2019 winner. Finally, a new method for maximizing non-differentiable functions is here suggested through reinforcement learning technique.

<table>
<thead>
<tr>
<th>Methods</th>
<th>PSNR</th>
<th>MS-SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>RL-SRGAN</td>
<td>22.34</td>
<td>0.783</td>
</tr>
<tr>
<td>SRGAN</td>
<td>22.15</td>
<td>0.780</td>
</tr>
<tr>
<td>LANCZOS</td>
<td>21.44</td>
<td>0.760</td>
</tr>
</tbody>
</table>

Table 1: The table shows the PSNR and MS-SSIM results obtained in the validation set for the proposed rl-srgan method with srgan and lanczos upsampling.

References