Obfuscated Human Faces Reconstruction

Daniele Mellino Edoardo Antonaci Alessandro Casalino
{daniele.mellino, edoardo.antonaci, alessandro.casalino.1}@studenti.unipd.it

Abstract

Modern image recognition methods based on artificial neural networks can recover hidden information from images protected by various forms of obfuscation. In this assignment we implement a basic architecture for image restoration and we compare it with two known model for super resolution,ESPCN and SRGAN, in order to test their abilities to deal with pixelated and blurred human faces. We test their behaviour using different metric and loss obtaining better results on starting pixelate faces than blurred ones.

1. Introduction

As user-contributed photographs and videos proliferate online social networks, video-streaming services, and photo-sharing websites, many of them can be found to leak sensitive information about the users who upload them, as well as bystanders accidentally captured in the frame. This awaken a privacy problem that led to the necessity of a solution. The problem however is that fully encrypting photos and videos before sharing or uploading them blocks direct information leakage, but also destroys information that is not privacy-breaching. An example of privacy protection technologies which aim to solve this challenge is facial obfuscation. We can see it everywhere online and in journalistic services. That seemed to solve the problem, although these technique do not provide well-defined privacy guarantees. In fact with the discover of recognition technique that involve deep learning it is possible to reconstruct the original faces, breaking the privacy of those subject.
In our work we create a simple CNN architecture for image restoration, after that we implement two networks, commonly used for super resolution: ESPCN and SRGAN and we train them in order to reconstruct previously obfuscated face images. The obfuscation techniques we adopt in the study are pixelation and blurring. We see that respect to the basic model, this methods behave better and better as far as the upscale factor increase.

2. Related work

It has been widely shown that CNNs and generative models are able to restore the information content of an obfuscated image enabling an accurate reconstruction [2]. At the same time there are model able of transforming a low resolution images into an High resolution one, increasing the resolution of a factor x16 (extreme super resolution) [9]. In the case of human face super resolution, state of the art approach are able to reflect face's details of both aligned and not aligned pose, through the use of progressive training and network aligning network.[ref 3]

In our work we study the capability of two famous SR model, ESPCN and SRGAN [5], [1], to deal with a restoration task.

Traditional evaluation metrics in super-resolution are *Peak Signal to Noise Ratio* (PSNR) and *Structural Similarity Index* (SSIM)[7]. However, these *fidelity* metrics are simple functions that measure the distortion to reference images and correlate poorly with the human visual response of the output. A high PSNR or SSIM does not guarantee a good looking results. For this reason we rely also on our eyes.

3. Dataset

In our study we start with the 'labeled face in the wild' dataset. It contains more than 13,000 images of faces collected from the web[1]. We use the deep-funneled version since it is said to give better results for face recognition. Those images contains not only the faces, but also some background. Therefore in the next step, we use a pre-trained model to recognize and filter only the faces. The cropped images have then different size. For that reason we resize them to (110,110). The size is chosen as the mean value of all the images size. The tool used to recognize the faces is not perfect, therefore we manually delete the images for which the tool failed. The results is then decomposed in 8.000 faces for training, 2,000 for validation and the rest are left for test. Another dataset used is the "Facesgrub" one [2] [8] : it contains about 100k HR actors and actresses'faces

[1] http://vis-www.cs.umass.edu/lfw/
[2] http://vintage.winklerbros.net/facescrub.html

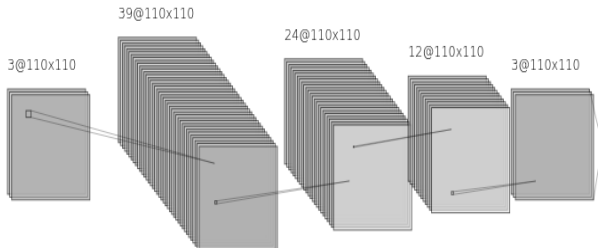

Figure 1. Basic model, for m=110

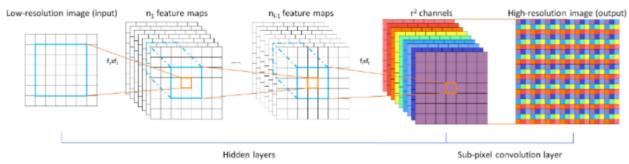

Figure 2. ESPCN

and pertinent background.In this case we used a script already tested to crop only the faces but also this time the attempt not always succeeded : a small part of the all photos were corrupted and so manually we discarded them to improve training procedure. Anyway,also in this case, the dataset has very huge size and we used only a small part of it : about 10k images splitted in 6k for training set and 4k for the test one.

4. Method

4.1. Architectures

The first architecture we build is resumed in Fig.1 It takes in input an m x m obfuscated RGB image and is expanded using 13 filters per channel, after which it gradually converges to less dimensions. For each convolutional layer we use ReLu activation function. We use this model as a baseline for more complex one, and we will refer to it as 'Basic Model'. The basic idea behind image super resolution is to add new pixels around and in between the existing one to reach the desired resolution. Those pixels can be for instance the weighted average of the nearby pixels (bi-linear or bicubic interpolation). The model we consider now, ESPCN, uses an improved technique called sub-pixel convolution. It consists in letting come, after some regular convolutional layers, a specific type of image reshaping called *phase shift*.
Image reshaping using phase shift is also called "pixel shuffle", which rearranges the elements of $H \times W \times C\dot{r}^2$ tensor to form $rH \times rW \times C$ tensor as shown below [3]. With this idea, the ESPCN architecture is simplified in Fig. 2.

The SRGAN architecture elaborated by Ladig et al. [1] is based on GAN model (generative adversarial network) where a generator G tries to reproduce an real image passed as input such that a differentiable function,the discriminator D , is not able to recognize and distinguish the artificial photo from the true one . This idea can be formalized using a min-max problem :

$$\min_{\theta_G} \max_{\theta_D} E_{I^{HR} \sim p_{train(I^{HR})}}[logD_{\theta_D}(I^{HR})] + E_{I^{LR} \sim p_{G(I^{LR})}}[log(1 - D_{\theta_D}(G_{\theta_G}(I^{LR}))] \quad (1)$$

where θ_G = $[W_{1:L}; b_{1:L}]$ denotes the weights and biases of a L-layer deep network and is obtained by optimizing a SR-specific loss function l^{SR}. I^{LR} is a low resolution image obtained during the training by applying a Gaussian filter to I^{HR} followed by a downsampling operation with downsampling factor r.

We describe I^{LR} by a real-valued tensor of size $W \times H \times C$ and I^{HR}, I^{SR} by $rW \times rH \times C$ respectively.

At the core of very deep generator network G, which is illustrated in Figure 3 are B residual blocks with identical layout. In the model is employed also the block layout proposed by Gross and Wilber [4]. Specifically, we use two convolutional layers with small 3x3 kernels and 64 feature maps followed by batch-normalization layers (useful to counteract the internal co-variate shift in deep networks) and ParametricReLU as the activation function. We increase the resolution of the input image with two trained sub-pixel convolution layers as proposed by Shi et al. [5]. To discriminate real HR images from generated SR samples we train a discriminator network. The architecture is shown in Figure 3. We follow the architectural guidelines summarized by Radford et al. [3] and use LeakyReLU activation (α = 0.2) and avoid max-pooling throughout the network. The discriminator network is trained to solve the maximization problem in equation 1. It contains eight convolutional layers with an increasing number of 3 x 3 filter kernels, increasing by a factor of 2 from 64 to 512 kernels as in the VGG network [6]. Strided convolutions are used to reduce the image resolution each time the number of features is doubled. The resulting 512 feature maps are followed by two dense layers and a final sigmoid activation function to obtain a probability for sample classification.

4.2. Perceptual Loss

The definition of our perceptual loss function l^{SR} is critical for the performance of our generator network. While l^{SR} is commonly modeled based on the MSE we design a loss function that assesses a solution with respect to perceptually relevant characteristics. We formulate the perceptual loss as the weighted sum of a content loss (l_x^{SR}) and an adversarial loss component :

$$l^{SR} = l_x^{SR} + 10^3 l_{Gen}^{SR} \quad (2)$$

[3]C is the channel color, H and W the height and width and r^2 is the number of channel.

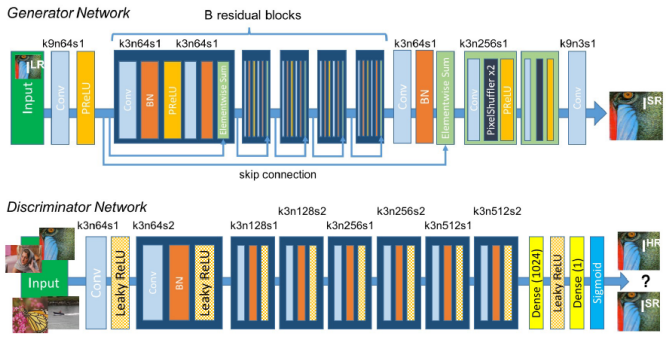

Figure 3. SRGAN

4.3. Pixel Loss

For our implementations we use two kind of losses: pixel and perceptual. Pixel losses evaluate the differences between each pixel of the output (or generated image) and the target one.

MSE Loss: Pixel wise Mean Square Error (MSE) is used during the training phase of the three models.It basically minimize the distance between the target image and the modelled one. Given 2 images (Y and $\hat{Y}$) of the same size $(m, n, 3)$ such that each pixel value ranges from 0 to 255. Then MSE is computed as following:

$$\mathcal{L}_{mse} = \frac{1}{n}\frac{1}{m}\sum_{i=1}^{n}\sum_{j=1}^{m}(Y(i,j) - \hat{Y}(i,j))^2$$

4.4. Adversarial Loss

: **Binary Cross-Entropy** (BCE) is used in the training procedure of the SRGAN model. In order to classify better and better the fake images created by the generator. The discriminator training loss $\mathcal{L}_{dsc}$ is computed as the sum the BCE loss $\mathcal{L}_{bce}$ of both batches (the one with the fake images(F) and the one with the original images(R)).

$$\mathcal{L}_{dsc} = \mathcal{L}_{bce}(F, l_{fake}) + \mathcal{L}_{bce}(R, l_{real})$$

where l_{real} and l_{fake} stand for images labeled as true or fake.

The total loss $\mathcal{L}$ of the generator is then computed as the sum of MSE loss between the generated images F and the target plus the BCE loss of the new images(considered as true).

$$\mathcal{L} = \mathcal{L}_{mse} + \mathcal{L}_{bce}(F, l_{real})$$

4.5. VGG Loss

VGG Loss is one type of *Perceptual Loss* used in style transfer and super-resolution. To use it, one has to choose one or more activation layers of a pre-trained VGG model. The most common version is the VGG-19.

To keep things simple, let assume we're interested only on one activation layer. We can define the VGG loss as the euclidean distance between the feature representations of the reconstructed image and the reference image. In formulas:

$$\mathcal{L}_{VGG/i,j} = \frac{1}{W_{i,j}H_{i,j}}\sum_{x=1}^{W_{i,j}}\sum_{y=1}^{H_{i,j}}(\phi_{i,j}\left(I^{HR}\right)_{x,y} - \phi_{i,j}\left(G_{\theta_G}\left(I^{LR}\right)\right)_{x,y})^2$$

where:

- $\phi_{i,j}$ is the feature map obtained by the $j-th$ convolution after the activation, before the $i-th$ maxpooling layer within the network
- $G_{\theta_G}\left(I^{LR}\right)$ is the reconstructed image
- I^{HR} is the reference image
- $W_{i,j}$ and $H_{i,j}$ describe the dimensions of the respective feature maps within the VGG network

4.6. Evaluation

As said above we evaluate our experiments through PSNR and SSIM.

PSNR, peak signal to noise ratio, measures the quality of a reconstruction in terms of absolute errors. The higher the PSNR the better the reconstruction, generally. In the case of this task, the output of the model is the reconstruction. Given the Mean Square Error between the target image and the output one,MSE, the PSNR is defined as:

$$PSNR = 20 \cdot log_{10}(\frac{max(I)}{\sqrt{MSE}})$$

Where $max(I)$ is the higher pixel value(in our case 1).

SSIM: is structural similarity, which measures perceived changes in structural information as well as in luminescence and contrast. The output of SSIM ranges between -1 and 1,with 1 indicating identical images.

5. Experiments

5.1. Basic Model

We train our basic architecture using MSE loss and Adam optimizer with starting learning rate equal to 0.001. We decrease the learning rate after 10 epoch of 0.002. The total number of epochs we use is 100.

We test our trained model for pixelated and blurred images obtaining the result in Tab. 1

As we can see from Fig. 4, even with this simple model we are able to deblur and unpixelate the images. We can notice that the result of the model is not very pleasent, it still seem obfuscated moreover details are not recognazable.

Filter type	Filter size	Avg PSNR	Avg SSIM
Blurring	6	23.37	0.64
Blurring	8	21.53	0.53
Pixelate	6	25.45	0.79
Pixelate	8	23.23	0.68

Table 1. Result obtained from the basic architecture for the 'labelled data in the wild' dataset

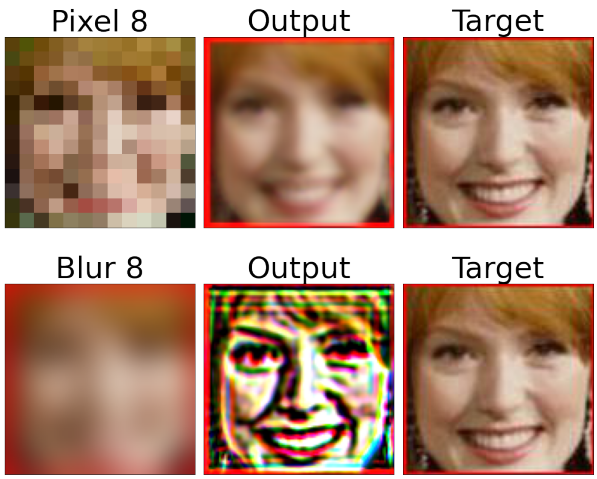

Figure 4. Results of Basic Model

5.2. ESPCN model

We tested ESPCN model with perceptual loss and in particular VGG loss. The use of this kind of loss alone doesn't give interesting results, therefore we decided to weight it in a linear combination with the Mean Squared Error loss. In this way, the computed loss in each iteration will be in the form: $k\,\mathcal{L}_{VGG} + \mathcal{L}_{MSE}$. Some interesting results has been found for $k = 10^{-3}$ and for smaller values.
Again, the images are resized to (110,110) shape before passing them to the model for training.
About the VGG model, We decided to use a pre-trained VGG-19 and its 35th feature layer for loss computations.
During train, we passed a filtered version of each image. The filter could be pixelation or gaussian blur, while the size of the filter could be 4, 8, or 16. We can notice from figures 5 and 6 that we visibly obtain better results if we train the model using pixelated images than using images with gaussian blur of the same size.

Upsampling	Avg PSNR	Avg SSIM
1x	22.51	0.51
2x	22.73	0.59
4x	23.87	0.66
8x	23.61	0.65

Table 2. Results using gaussian blur with size = 8, batch size = 64, learning rate = 10^{-3}

The model's output looks visibly better when we use a

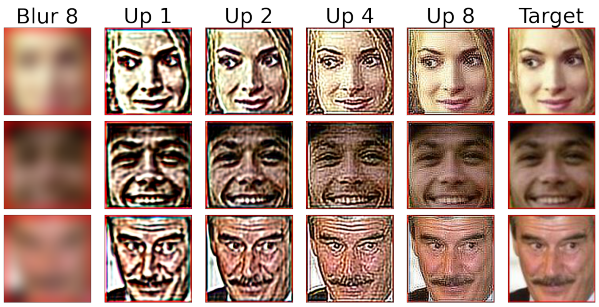

Figure 5. Results of a ESPCN model trained with perceptual loss + MSE loss, with $k = 10^{-3}$. The blur filter on first columns has kernel of size = 8, while on the next columns we progressively increase the upsampling value. In the last column we show the images without any filter

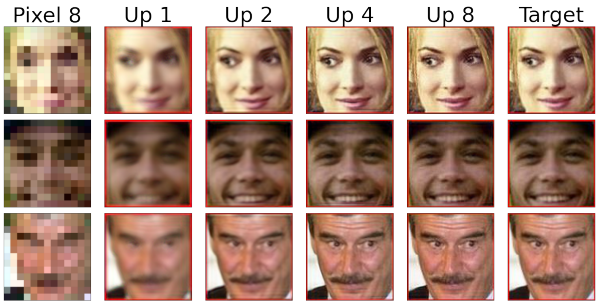

Figure 6. Results of a ESPCN model trained with perceptual loss + MSE loss, with $k = 10^{-3}$. The pixelation filter on first columns has kernel of size = 8, while on the next columns we progressively increase the upsampling value. In the last column we show the images without any filter

Upsampling	Avg PSNR	Avg SSIM
1x	18.39	0.36
2x	19.76	0.42
4x	20.44	0.46
8x	17.02	0.25

Table 3. Results using gaussian blur with size = 16, batch size = 64, learning rate = 10^{-3}

Upsampling	Avg PSNR	Avg SSIM
1x	23.45	0.70
2x	23.62	0.70
4x	24.09	0.73
8x	23.74	0.71

Table 4. Results using pixelation filter with size = 8, batch size = 64, learning rate = 10^{-3}

pixelation filter instead of a gaussian blur and this is confirmed by PSNR and SSIM scores. The best scores for both PSNR and SSIM are for pixelation filter with size = 8 and upsampling 4x. This could interpreted as the fact that, for higher upsampling value, the model searches for too many details compared to the original image. Having said that,

Upsampling	Avg PSNR	Avg SSIM
1x	19.23	0.44
2x	19.54	0.46
4x	20.04	0.50
8x	20.43	0.52

Table 5. Results using pixelation filter with size = 16, batch size = 64, learning rate = 10^{-3}

Figure 7. Comparison results of MSE loss alone versus MSE loss + VGG loss

fixing the size of both gaussian blur and pixelation, and for all the upsampling values considered, the scores are comparable between each other, although we can visibly see some differences because of how our recognition as human beings works.
Comparing to ESPCN with MSE loss only, we have better results if we add perceptual loss. For example, using the same parameters of the previous cases, for an upsampling 8x and an input blurred with filter size equal to 8, MSE loss alone obtain Average PSNR = 21.10 and Average SSIM = 0.54, while if we add the perceptual loss we obtain Average PSNR = 23.61 and Average SSIM = 0.65. Although the differences are quite noticeable and demonstrate the strength of perceptual loss, it's difficult to visually see the improvements. An example is in Fig. 7

5.3. SRGAN model

Different experiments (see table 6) are also done using SRGAN model on two datasets : the best is for 'labeled face in the wild' using an upscaling x4 and it has brought to PSNR $\sim$ 33 dB and SSIM $\sim$ 0.956 with lr = 0.001 and batch size = 2. For super resolution and unpixelate task we achieved interesting faces reconstruction as we can see in figure 9. Unfortunately for the unblurred task the network is not able to make significant upgrades.

learning rate	batch size	average PSNR
10^{-3}	5	14
10^{-4}	10	30
10^{-3}	5	32
10^{-4}	1	14
5*10^{-5}	5	32
10^{-2}	5	14

Table 6. SRGAN trials

During the training procedure was applied a sort of "early stopping" based on the PSNR and SSIM's behaviour

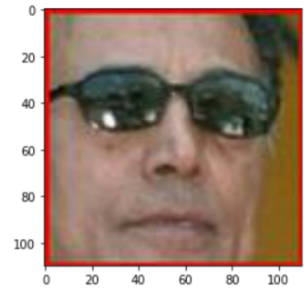

Figure 8. target photo

Figure 9. From left to right: blurred(r=3) and pixelate(p=4) to results

: if after 5 epochs (fixed taking in account the time-consuming GPU) at least one of them didn't increase we stopped the training procedure and save the parameter of the model. Two observations must be done :

1. using blurred images or pixelate ones the capability of the network to recostruction images does not improve.
2. about "Facesgrub" dataset : the high-quality of the images was not be very useful for SRGAN model because almost all the images have a lot of pixels (also 1600 x1600) and during training procedure we resize them and so some informations are lost.

6. Conclusion

In this project we explored the potentialities offered by deep learning to face a common problem in computer vision: images reconstruction. Starting from uncorrupted images we trained two different models, ESPCN and SRGAN, obtaining different results.
These results depend both by starting images with different resolutions (LR for "labeled face in the wild" and HR for "facegrub") and by task fixed as aim: blurration or pixelation.

Looking at results for both of them we can state that ESPCN is better than SRGAN in reconstruction task: improving the "upsample" from 1 to 8, very blurred/pixelated images recover their original form.
This is true even if PSNR and SSIM are not excellent if compared with results obtained training SRGAN: that is another proof that shows how these parameters don't affect uniquely the results.
Indeed this model is able to reconstruct quite fine the starting images when those are pixelated but it does almost nothing when input ones are blurred. This is true for both datasets without difference if the input images are LR or HR. Both models outperform the performance of the Basic model as one can see from the results. A possible improvement could be using HD dataset with a more large "crops" photos as inputs: using Colab GPU we have time and size limited use and so we are forced to limite our trials both for the number of photos in training/test and also their shape. In this hypothesis SRGAN could improve its performance because it is a more complex architecture than ESPCN and so more weights must be properly setted. We would like to notice that in this work we do not use special attention loss able to consider the facial structures, it would be an other possible development of our study.

References

[1] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew P. Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. *CoRR*, abs/1609.04802, 2016.

[2] Richard McPherson, Reza Shokri, and Vitaly Shmatikov. Defeating image obfuscation with deep learning. 09 2016.

[3] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2016.

[4] Michael Wilber Sam Gross. Training and investigating residual nets. 2016.

[5] Wenzhe Shi, Jose Caballero, Ferenc Huszár, 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, 2016.

[6] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[7] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[8] Hong wei Ng and Stefan Winkler. A data-driven approach to cleaning large face datasets.

[9] Kai Zhang, Shuhang Gu, Radu Timofte, and Taizhang et al. Shang. Ntire 2020 challenge on perceptual extreme super-resolution: Methods and results. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2045–2057, 2020.