PROJECT
IMAGE DENOISING COMPARISON MODELS

Submitted by : Di Luccio Luca,
Faiella Simone,
Trerè Marialuisa,
Jan. M Tanveer

Supervisor: Luigi De Maio
Assistant Professor
Department of Informatica

Presented to : Michele Nappi
Professor
Department of Informatica

University of Salerno
Department of Informatica

Introduction

The project that has been developed aims to eliminate noise on faces through neural networks. In addition to denoising, an assessment was performed on how the neural network works on a completely cut out face compared to a subject with a background behind it.
When we talk about image noise, we refer to a variation of brightness or color information in images, which causes an undesired visual quality. The image denoising is therefore a process of removing such artifacts from an image.
Obviously the noises that an image can present are multiple, each with certain characteristics, it is therefore appropriate to carry out a careful analysis of the type of noise to be able to obtain an optimal result. Some examples we can find:

- ***Salt and pepper***:
  This noise can be caused by sharp and sudden disturbances in the image signal. It presents itself as sparsely occurring white and black pixels;
- ***Gaussian noise***:
  The main sources of Gaussian noise in digital images arise during acquisition e.g. sensor noise caused by poor lighting, high temperature, transmission or, obviously, a combination of them;
- ***Speckle*:**
  It is a "noise" granular that inherently exists and degrades the quality of active radar, synthetic radar openings (SAR), images of medical ultrasound and optical coherence tomography.
  It is usually present in webcams and video surveillance cameras and for this reason it is very common.

What approaches can be used to solve the problem?

Although it is a common problem, the approaches to the denoising problem are purely theoretical, leaving out everything concerning the actual implementation, or possibly omitting it. It is possible to resort to different approaches, each one differs from the other and it is good to pay attention to the objective you are focusing on to obtain an optimal result:

- ***Filters***:
  This approach use different algorithms to remove the noise, but they need to know the type and intensity of the noise in addition to the degradation function;
- ***AutoEncoders***:
  To build an autoencoder, there is need of three elements: an encoding function, a decoding function, and a distance function between loss, compression and decompression (i.e. a "loss" function).
  It is of fundamental importance to emphasize that autoencoders are lossy, which means that the decompressed outputs will be degraded compared to the original inputs, which is why, after careful analysis and verification, this approach has been discarded due to the type of data with which the project has been dealt with.
- ***Convolutional Neural Networks****:*
  CNNs are similar to normal neural network and are commonly used for visual image processing. CNNs are formed by input layers, various hidden layers and output layers. It works like a hierarchy and each layer is made up of a set of features extracted from the original image, each feature has its own weight and different features can be combine back to get the original image.

- ***Generative Adversarial Neural Networks****:* This type of approach is completely new, having been born in 2014.
  With GANs a random noise is input to a Generator, a neural network, this one generate fake images with the noise, these images are input to a Discriminator, a neural network, together with the original images of the dataset. Then, the Discriminator is trained until it can recognize the original image from the fake one. This method is one of the best approaches in the generation of images as the output has to fool the discriminator network to get validated, however it is not widely used because of the slow nature of the training

Regarding the project, it was not limited to a simple search for the fastest and most intuitive solution. The final result is based on an evaluation of the different techniques shown in the paper, often shown in an exclusively mathematical way.

Libraries used

There were many libraries used to complete the project, among the fundamentals we can identify:

- ***Keras***:
  This is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. It was developed with a focus on enabling fast experimentation. This library is used to build and train the model, which is the fulcrum of any project with neural networks;
- ***TensorFlow***:
  It is an open source software library for high performance numerical computation. Its flexible architecture allows easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs);
- ***Dlib***:
  Dlib is a software library written in the programming language C++. Its usefulness in the project can be found above all in the cropping script of the image within which the face is detected for a more accurate application of the denoising;
  We use Dlib to crop in the most accurate way possible the images, by using the hi-speed facial detection functions it provides
- ***Matplotlib***:
  It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits. It is used to generate, display and save graphs of different nature about the models we build and train. It is therefore possible, thanks to it, to visualize the result of the work carried out;
- ***NumPy***:
  It guarantees support for multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays;
- ***OpenCV***:
  This library has been used for image editing, as it provides considerable support in this field, indeed it is a library of programming functions mainly aimed at real-time computer vision;
- ***Scikit-image***:
  This is a collection of algorithms for image processing; for this reason it was used for noise creation and for image reading and processing.

First operations

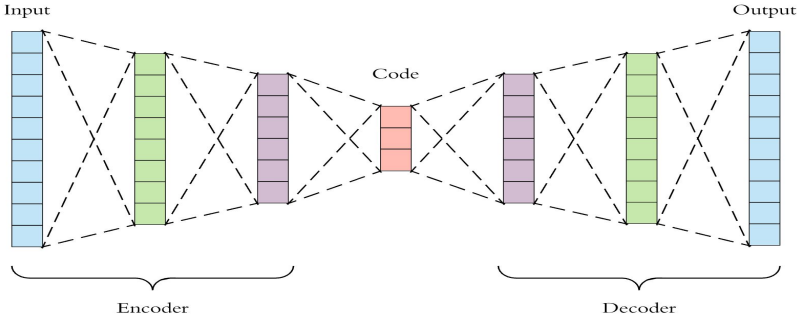
The dataset that was used for the development of the project is characterized by 200,000+ celebrity images. Obviously, being a generic database and not properly related to the goal of denoising, a series of changes have been made so that it could be useful for the development of the project. The images have been cut out through the use of the dlib library, using the facial recognition it was possible to acquire only the most important information. The faces obtained were then aligned so as to help the neural network to perfectly understand the focal points of the face, in order to facilitate denoising.
Working with a database of this size, or even in the order of thousands of images, is absolutely expensive, and for this reason it is advisable to make the appropriate configurations so that it is possible to work with the GPU. In particular for the project, it has relied on a PNY nVidia Quadro P4000 8 GB Graphics Card, which has allowed to reduce development time compared to the very long time of a less advanced video card.
Although relatively high computational power has been used, 64x64 images should still be used, as a larger size would have greatly affected performance, both in terms of time and the computer's ability to complete the operation.
Following this logic, the number of images to train the models has been drastically reduced to about 10,000, which, after numerous tests, proved to be the optimal number in order to obtain positive results with certainly reduced times.
"*Speckle*" type noise was manually added to the images. The choice of this type of noise is dictated by two important evaluations: first of all being the project linked to smart objects such as surveillance cameras, it is the most suitable for the project objective. There are also many papers that use this noise. It is also worth pointing out how the last paper evaluated, through which the appropriately modified model was extrapolated, based its functioning on a Gaussian type noise, so it should easily adapt to the latter too.

Types of models used

As already anticipated, *multiple approaches* have been analyzed and used, the results of which have been kept for educational purposes, in order to show the differences that characterize them in a practical way.

AutoEncoders:

Precisely because of the defects already mentioned above, although it has however been analyzed and tested in practice, the use of the autoencoder for the final solution has been discarded. Because of the intrinsic structure that characterizes them they are not the best solution for color images and relatively large. In detail they will be able to work well for small images (in the order of 28x28) and in black and white, so that the loss of information can not heavily affect the final result.



What derives from the use of this model matches what are the theoretical evaluations carried out, and can be observed through these empirical results:

As the quality of the images begins to increase, due to the intrinsic structure of the neural network, obtaining only the information of the images that are considered important, they tend to be blurred.

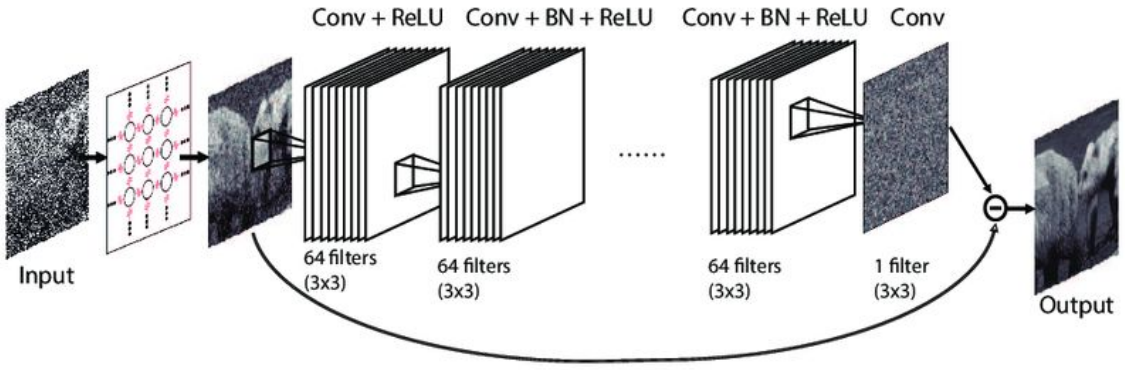
The structure of this neural network has been analyzed through the study of this paper.

DNCNN:

Based on the study of this paper, an architecture based on DNCNN was analyzed and tested. In this architecture it is possible to find three types of layers, with three different colors.

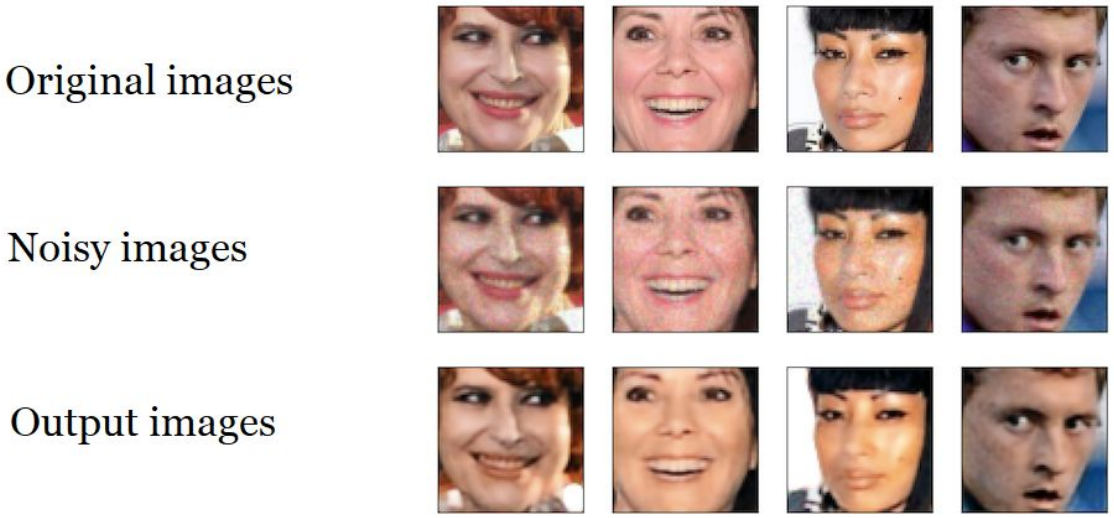
1. Conv+ReLU: for the first layer, 64 filters of size $3 \times 3 \times c$ are used to generate 64 feature maps. Here c represents the number of image channels, i.e., $c = 1$ for gray image and $c = 3$ for color image.
2. Conv+BN+ReLU: for layers 2 until $(D - 1)$, 64 filters of size $3 \times 3 \times 64$ are used, and batch normalization is added between convolution and ReLU.
3. Conv: for the last layer, c filters of size $3 \times 3 \times 64$ are used to reconstruct the output.

To sum up, this DnCNN model has two main features: the residual learning formulation is adopted to learn R(y), and batch normalization is incorporated to speed up training as well as boost the denoising performance. By incorporating convolution with ReLU, DnCNN can gradually separate image structure from the noisy observation through the hidden layers.



The use of this model went through two phases: initially it was tested with 64x64 images in black and white. Indeed, the models have been previously evaluated with this type of images for a preventive and faster evaluation of the goodness of the neural network.

As can be seen, the results obtained are more than satisfactory for images of this kind, and therefore it has been found appropriate to carry out an evaluation also with the respective color variations.



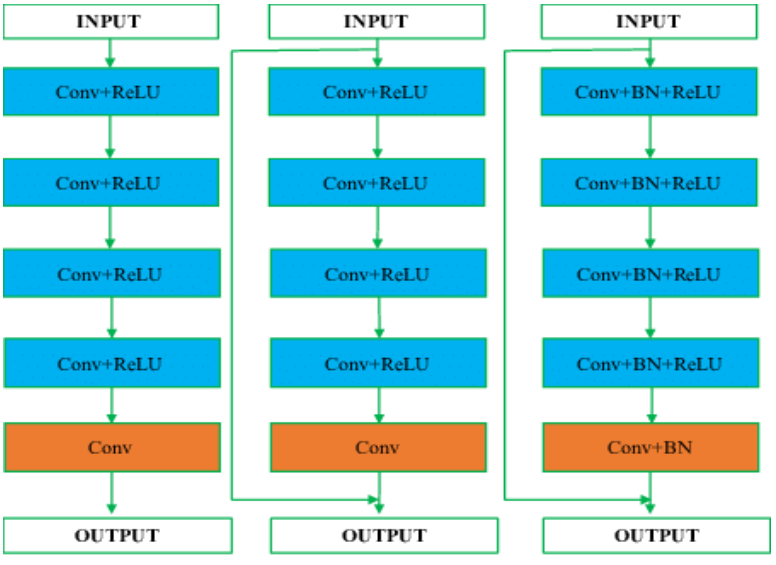
The results regarding color images, although with this model are starting to be valid have not fully met the expectations, the images are still losing quality, although not to the levels of autoencoders.
It was therefore decided to evaluate alternative solutions, in order to find the best solution to the problem.

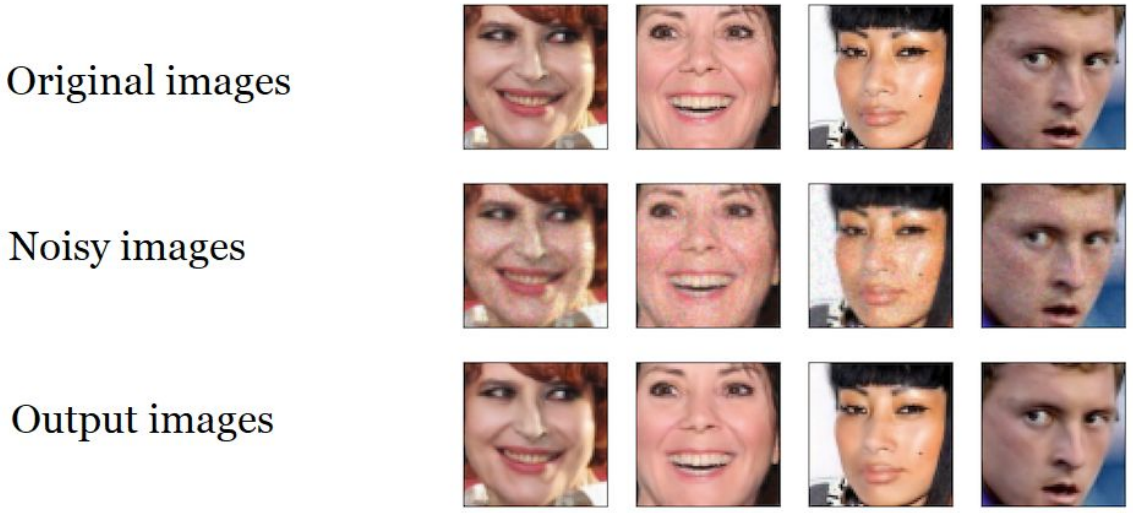
WIN5-RB

WIN5, WIN5-R and WIN5-RB, three models analyzed to solve the problem of denoising, have the identical basic structure: L = 5 layers and K = 128 filters of size F = 7×7 in most convolution layers, except for the last one with K = 1 filter of size F = 7×7. The differences among them are whether batch normalization (BN) and an input-to-output skip connection are involved.
WIN5-RB has two types of layers with two different colors:

1. Conv+BN+ReLU : for layers 1 to L – 1, BN is added between Conv and ReLU.
2. Conv+BN: for the last layer, K = 1 filters of size F = 7 × 7 is used to reconstruct the R(y) ≈ –n.

In addition, a shortcut skip connecting the input (data layer) with the output (last layer) is added to merge the input data with R(y) as the final recovered image.

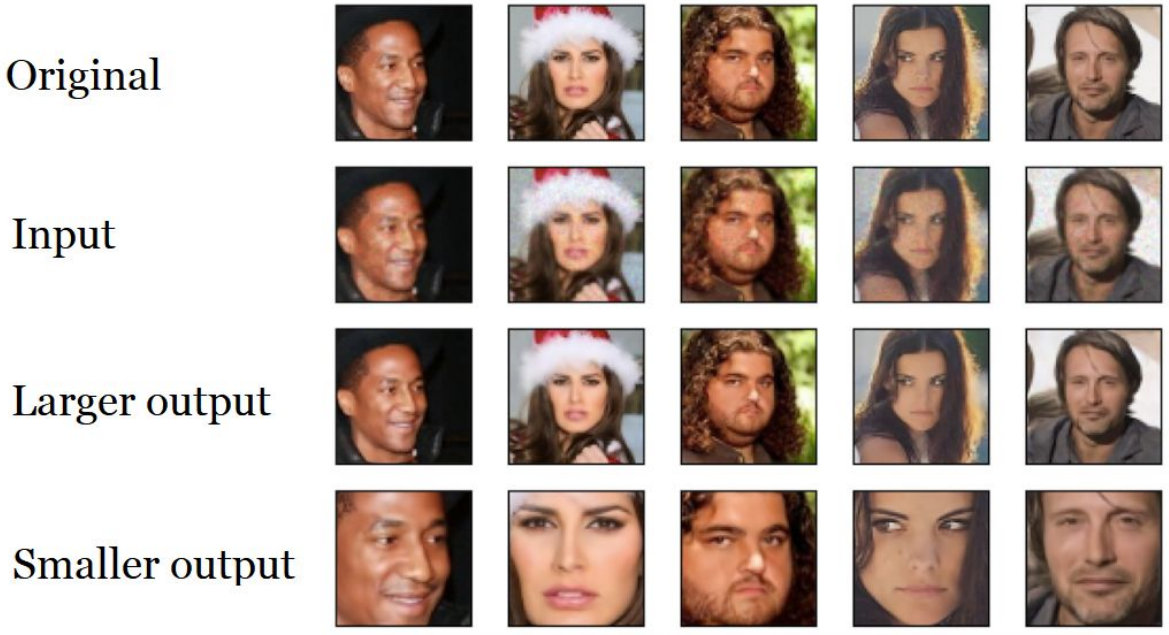




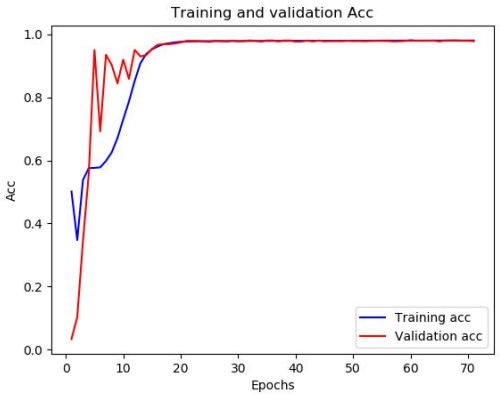
The result given by this model is appreciable, the neural network works correctly both for black and white images, both in color, through a training of 10,000 images.
The study of these methodologies was made possible thanks to the study of this paper which explains in detail how the neural networks based on WIN5 work.

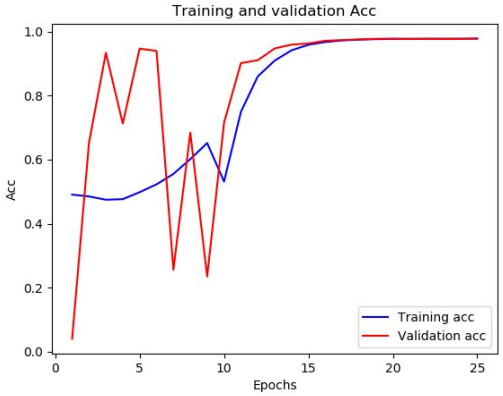
Evaluation step

To meet the final goal of the project, a new dataset was created, in which images with faces are present with a background, in order to evaluate what are the differences in performance and results of the neural network compared to an image in which the face is cut out perfectly.



Although the neural network is adapted to face with a background behind it, surprisingly positive, because both achieve high accuracy values, it is important to emphasize that the best performances are obtained on completely cut faces, as the presence of faces only helps the network neural to optimally differentiate the presence of noise compared to the faces and thus manages to converge to an optimal situation in a simpler way.





The above graphs, as it is possible to see, show the difference in behavior between the two types of models analyzed. The first shows the behavior of the network with images without background, while the second shows the same network with images with background behind a face.

References

- Image Denoising with Deep Neural Networks by Aojia Zhao, Stanford University, September 2017.

- Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising by Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang, August 2016

- Learning Pixel-Distribution Prior with Wider Convolution for Image Denoising , Peng Liu, Ruogu Fang. July 2017.