Image Inpainting Using Pre-trained Classification-CNN

Yaniv Kerzhner
Depaertment of Electrical Engineering
Technion, Haifa 32000, Isreal

Adar Elad
Depaertment of Computer Science
Technion, Haifa 32000, Isreal

Yaniv Romano
Depaertment of Electrical Engineering
Technion, Haifa 32000, Isreal

Abstract

Image inpainting is an extensively studied problem in image processing, and various tools have been brought to serve it over the years. Recently, effective solutions to this problem based on deep-learning have been added to this impressive list. This paper offers a novel and unconventional solution to the image inpainting problem, still in the context of deep-learning. As opposed to a direct solution of training a CNN to fill-in missing parts in images, this work promotes a solution based on pre-trained classification-oriented CNN. The proposed algorithm is based on the assumption that such CNN's have memorized the visual information they operate upon, and this can be leveraged for our inpainting task. The main theme in the proposed solution is the formulation of the problem as an energy-minimization task in which the missing pixels in the input image are the unknowns. This minimization aims to reduce the distance between the true image' label and the one resulting from the network operating on the completed image. A critical observation in our work is the fact that for better inpainting performance, the pre-training of the CNN should be applied on small portions of images (patches), rather than the complete images. This ensures that the network assimilates small details in the data, which are crucial for the inpainting needs. We demonstrate the success of this algorithm on two datasets: MNIST digits and face images (Extended Yale B), showing in both the tendency of this method to operate very well.

1 Introduction

Inpainting is the process of reconstructing missing parts in images [8]. This problem started in the fine arts, occupying many artists since the middle ages, working on restoring painting in museums and elsewhere. More recently, the problem and its applications were presented in their digital form by Bertalmio-Sapiro-Caselles-Ballester (SIGGRAPH 2000), this led to a long series of publications (e.g. [9, 10, 11, 13]), offering various novel ideas for handling the inpainting task. This problem remains prevalent today and the most recent comers to this activity are neural networks.

A common approach for handling image inpainting by deep learning tools is to train a network to tackle directly the problem (e.g. [1, 3]). In such a case one trains the network based on relevant and direct information, being pairs of images: images with missing parts of information and their complete versions. The key idea is to train the network to close the gap between input and output, and then hopefully generalize to other images and other hole shapes.

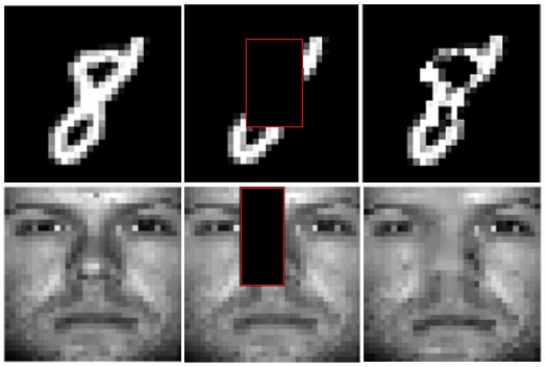

Figure 1: Example of the proposed inpainting. The original image (left), the image with the hole (middle), and the restored result (right).

The approach we present in this paper is also based on deep learning, but it is different from the common path, because our solution leans on a CNN that has been pre-trained to perform classification. When inpainting an image, the

proposed algorithm uses this CNN to predict the missing content in the image and obtain an estimate that best suite the known classification we aim for. For example, when inpainting an image of the digit 5 (taken from the MNIST database), the algorithm is initialized with zero gray level values in the hole area, therefore the network will either fail to identify its class, or succeed with a weak confidence. Then, the proposed algorithm turns to minimize the energy of the classification error, changing the content in the hole area to make the identification more certain.
In fact, there are two presumptions in this work: (1) the CNN is assumed to have memorized the objects it has been trained on, and thus could serve the inpainting task; and (2) the hole in the image must impair the original classification, as otherwise the network responsible for filling in the missing data will simply "refuse" to inpaint. Throughout this paper we come back to these assumptions showing that training of the CNN properly is key in the success of our scheme. Also, we show that the size of the hole is critical for getting successful inpainting images.
The proposed algorithm works by activating two forces - a force that we have already introduced, which pushes the missing content to improve the identification, and a second force that strives to achieve an image that looks natural. The proposed algorithm uses an energy function consisting of a linear combination of these two penalties. The minimizer of this energy function is the inpainting result. In order to find this minimum, we follow the method of gradient descent and employ backprpopagation. As for forcing the naturalness of the inpainted content, we suggest using the Total-Variation to ensure that the result is smooth and rimless at the edges of the hole.
During the development of the proposed solution we tried to avoid changing the CNN that we use, because at the center of our work stands the statement that there is no need to create a special network in order to tackle the problem, but rather using an existing one. However, we noticed that training the network in a slightly different way could significantly improve the results. More specifically, we found that training on portions of images rather than the whole images allows the network to classify while simultaneously better assimilate the local visual characteristics of each object. Technically, this was achieved by using the same training set, feeding each example many times, masked (by zeroing) at random so as to remove large portions from it. Overall, this training improved the performance of the inpainting results, while keeping the overall recognition rates the same.
In the results section we present two sets of experiments. We first inpaint images of digits by a CNN that was trained to identify them (trained on the MNIST database). The second experiment focuses on inpainting of face images by a CNN that was trained to identify different people (trained on the Extended Yale B database). In both cases the proposed algorithm is shown to be able to solve the problem for various holes. We also present failure cases, which allow us to reflect on the weaknesses of the proposed scheme.

2 Related Work

While there are various methods for tackling the inpainting problem, herein we focus on the ones that rely on neural networks. The work reported in [1] proposes a blind inpainting method based on a fully convolutional neural network, trained on a dataset of corrupted/ground truth sub image pairs. Once trained, this network was shown to both detect the corrupted region in the image and inpaint the missing content with high levels of success. [2] follows a blind inpainting approach based on a pre-trained denoising auto-encoder. Both [1] [2] handle the inpainting problem by directly training a neural network on example-pairs of patches/sub-images of corrupted images and their ground truth. Interestingly, both algorithms rely on the idea of training their networks by using sub-images rather than whole images for the inpainting task. While our approach is very different from the above methods, we found that this feature is critical to the success of our method as well. We shall note that one of the limitations of approach taken in [1] is the need for a new training procedure for every new type of corruption.
By analogy to auto-encoders offered in [2], Deepak Pathak et al. [3] presented a novel approach with context encoders for the inpainting task. The context encoder allows to add missing content to the corrupted regions based on the surroundings of the missing parts and the context of the image. The training procedure of the context encoder aim to minimize a pixel-wise reconstruction loss and an adversarial loss. The reconstruction L_2 loss is responsible for filling the overall structure of the missing regions with regards to its context. The adversarial loss on the other hand, trying to make the out-

put image look as realistic as possible. Both works [3] [4] share in common the idea of using a function with two loss components - one for reconstruction and the other for the naturalness of the image (based on adversarial networks). A trace of this idea appears in our work as well, but in an entirely different way, the proposed cost function consists of two penalties - the first considering the reconstruction based on the network's visual knowledge, and a second, total-variation penalty responsible for the naturalness of the result.

3 Proposed Method

Differently from the above-described papers, our solution is not based on a network that has been trained to inpaint (either directly as in [1, 2] or indirectly as in [3] [4]). Instead we present the concept of using a classification-oriented network for solving a completely different problem based on the data it has been trained to handle. In common with several recent papers (e.g. [5]), our algorithm is founded on methods of optimization relying on the CNN's knowledge. Indeed, other inpainting algorithms (e.g. [1, 2, 3, 4]) could use the penalties we present as an additional force to improve their results.

3.1 The Energy Function

The algorithm we present inpaints a given image using an energy function, which is to be minimized with respect to the unknown pixels in the hole. This cost function is formulated as a linear combination of two penalties. The first relies on the pre-trained classification network, computing the distance between the given (and desired) label, and the one computed on the input image:

$$P_{content}(X) = \| \, CNN(X) - \ell \, \|_2^2 \, .$$

In this expression, $X \in R^(N \times M)$ is the image we inapint, $CNN(\cdot)$ is a given classifier, and ℓ is the desired classification for completion. As such, the purpose of this expression is to promote the image X (or more accurately, change the missing pixels in this image) to be classified as an image that belongs to the class ℓ, as determined by the CNN. The second penalty in the overall energy function aims to refine the estimated content in order to make the reasulting image more natural. This penalty is formulated as the Total-Variation:

$$P_{TV} = \| \, M_{TV} D_h X \, \|_1 + \| \, M_{TV} D_v X \, \|_1 \, \cdot$$

where D_h and D_v are matrices of size $NM \times NM$, denoting the horizontal and vertical first derivativs, and $M_{TV} \in R^{NM \times NM}$ is a mask that allows us to determine which parts of the image we want to effect. In our algorithm M_{TV} is detemined either as the whole image (works best for digit inpainting), or a region surrounding the hole (works best for face inpainting).

3.2 Algorithm

This challenging minimization is done using the Gradient Descent algorithm. Observe that the derivative of $P_{content}$ calls for the computation of the derivative of the CNN with respect to the input image, implying that in each iteration we use a back-propagation process, so as to propagate the label error to the input image domain. Algorithm 1 describes the proposed algorithm. The observant reader will find this to be somewhat related to the work in [7] and others, where a similar optimization was used for stylizing images. Our approach is also reminiscent of Vedaldi's work [5] in which he analyzed the behavior of given networks by optimizing with respect to their inputs.

Algorithm 1: Gradient Descent

Data: Y (the input image), ℓ (the desired lable), M (the mask), CNN (a classifier)
Result: Inpainted X
Initialize X in the missing parts;
while $\| \nabla E(X) \| > \epsilon$ **do**
  calculate: $\nabla E(X) = \nabla_X P_{content}(X)$
    $+\lambda(D_h^T M_{TV}^T sign(M_{TV} D_h x)$
    $+D_v^T M_{TV}^T sign(M_{TV} D_v x))$
  update X: $X = X - \mu \nabla E(X)$
end

3.3 Initialization

Since we are aiming to minimize a non-convex function, we face an uncertainty in the minimum we reach. This optimum depends on the initialization. In order to improve our chances of converging to a meaningful minimum that serves well the inpainting task, we should initialize the missing parts wisely, as we describe below. This provides some control over the resulting estimate. Here is an illustration of the uncertainty in the minimum we reach, obtained for different initializations:

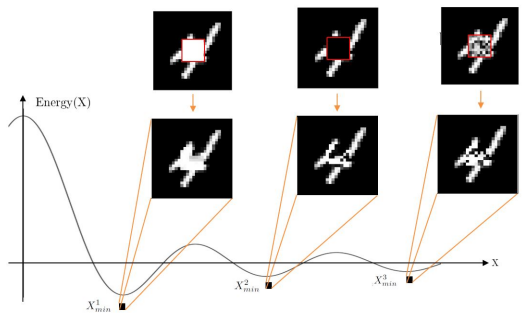

Figure 2: An illustration of the uncertainty in the minimum we reach. Each initialization leads to a different inpainting result.

The initialization methods we have explored include:

(i) Fill-in the missing parts using linear interpolation. In this way we create a diffusion of the hole's boundaries into the missing parts (this initialization worked best for digit inpaintings).

(ii) Fill-in the missing parts with the average-image of the desired class/label. This serves as a successful initial guess for the classifier (this initialization worked best for face inpaintings). It is important to note that for all the results, the inpainted image does not match the average at parts that were missing. This means that the algorithm uses the average-image only for an initial guess.

Figure 3 : Initialization strategies for the algorithm. The right-most is the corrupted image to inpaint. The left-most is initialized with the average image referring to all the examples sharing this label. The middle image presents a diffused initializetion.

3.4 Improvement Techniques

In this section we discuss several techniques that significantly improved our results. The first of these consists of randomly shifting the temporary image before updating it. More specifically, before line 3 in Alorithm 1 we shift randomly X in each axis and after line 5 we shift X back to its previous location. This feature helps the network identify objects in an image without depending on their location. In our context, this allows the network to add new content to the corrupted regions. This method was inspired by [5].

A second idea that was found highly influencial is patch-based training. We found out that pre-training the CNN on portions of images, rather than the complete ones leads to sunstantial improvement in the overall inpainting performance. The patches we refer to are obtained by multiplying the original images by a random mask that nulls all content except a small rectangular region. This approach drives the classifier to be sensitive and memorize small pieces of the image content. In our experiments, we found that by doing so, the network is able to assimilate better the visual characteristics of each label.

On top of the above two ideas, another beneficial force that we have found as usefull in the context of MNIST is to push the destination image into binary values. We formulated an additional penalty loss function to achieve this goal. This loss function was set as high for intermediate shades (i.e. away from black or white), while being low for the 0 and 255 gray values.

4 Results

In this part we demonstrate the proposed inpainting algorithm and examine the importance of its inner details. Our experiments are divided into two parts, the first testing the inpainting on digit images (taken from the MNIST database), and the second focuses on face images (using the Extended Yale B database). In each of these parts, all the inpainting results shown use the same energy function, the same optimizaiton technique, the same pre-trained CNN, and the same parameters. This testifies to the strength and the robustness of the proposed scheme to the type and the amount of the missing data in the images needed to inpaint.

Alongside the display of successful inpainting results, we also show failure cases, and discuss their implication on the proposed algorithm. Also, we provide experiments in which we vary key parameters or decisions in our scheme, in order to assess their influence on the results obtained.

4.1 Inpainting Handwritten-Digit Images (MNIST)

The images shown in Figure 4 are successful inpainting results for images of digits, taken from the MNIST database. The CNN used in these experiments was trained to classify digits, and based on the LeNet architecture [14]. However, as opposed to the usual training of such networks, which uses the full 28×28 pixels images, we trained the network on masked versions of the original images. More specifically, each training image was used 5 times, each time masked differently while exposing a portion of size 7×7 to 20×20 pixels (at random location) and zeroing the rest. This way we encourage the classifier to be sensitive to all the local details, a fact that proved beneficial for the inpainting. As shown in Figure 4 the inpainting results are nearly perfect, even if they may differ from the original content in the hole.

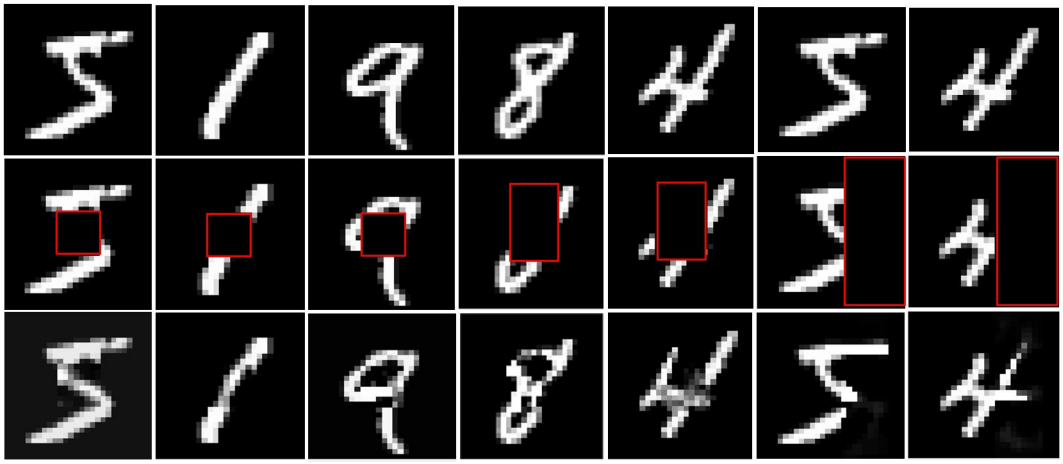

Figure 4: Each column represents a different result. The ones in the first row are the original images. The images in the second row are the corrupted versions of the ones in the first row, masked with different types of holes. The images in the third row are the results of our inpainting.

4.2 Inpainting Face Images (Yale Extended B)

Figure 5 illustrates the results we obtain for facial images. Unlike the handwritten-digit case, we use of the Jitter technique, as explained in Section 3.4. The CNN used in these experiments was trained to classify people by portrait images, and based on the LeNet architecture [14]. Similarly to the method used for digit inapinting, we trained the network on portions of the database images. We noticed that many images are relativly dark, therefore many portions of these images are simply lacking of meaningful data. In order to solve this, areas that suffer from this problem (mesured by the energy of the patch) were replaced by the whole original image.
Note that in some of the results, the estimated content does not match the one of the original image. These are interesting results since we visualize what the network itself considers as an image that belongs to a specific class.

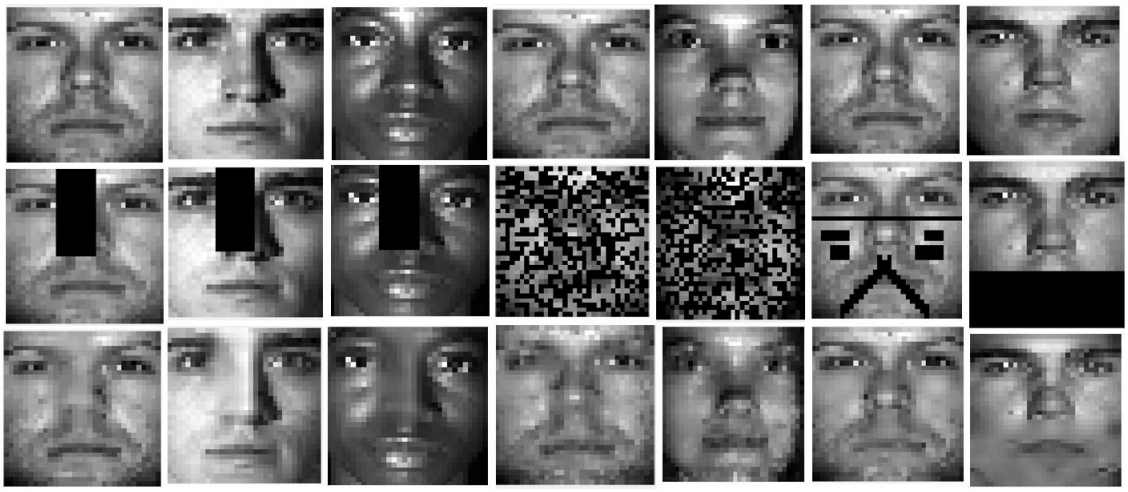

Figure 5: Each column represents a different experiment. The images in the second row are instances of the original images of these persons. In the third row, the corrupted images with missing data are presented, and the most bottom row displays the inpainting results.

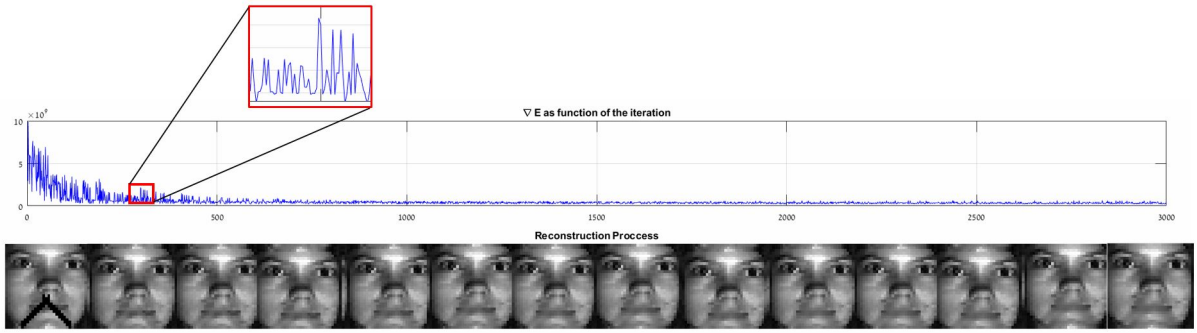

Figure 6: The graph shown is the norm of the gradient of the energy function versus the iterations of the gradient descent. Below this graph are the inpainting result for every 250 iterations. As can be seen, the gradient norm converges to zero. Moreover, by zooming, it is possible to see that the Jitter effect in each iteration produces a pick in the gradient caused by the motion of the image.

4.3 Failure Cases

The algorithm we presented is repleted with various parameters and techniques that aim to improve the way the missing parts are inpainted, both in terms of content and in terms of naturalness of the result. This section presents the importance of some of these parameters and techniques with results illustrating their effect.

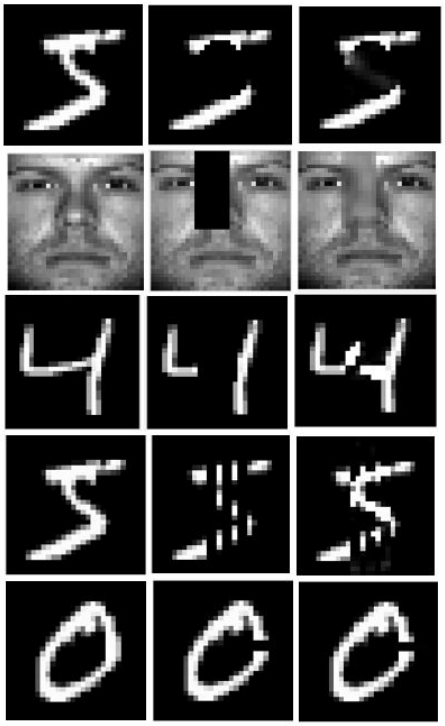

Figure 7: The left column displays the original image, alongside is the corrupted image, and on the right column is the inpainting result. Every row demonstrates a different change in the ideal algorithm.
In the first row we show the inpainted image, obtained by a classification network trained on the whole images rather than portions. As can be seen the inpainting is not done properly because the network is not sensitive to the visual details.
In the second row we present the inpainting result for the parameter modification responsible for the image smoothness and naturalness. This parameter takes part in the energy function and cosmetically affects the inpainting result.
The third row displays the result of the original algorithm with a different initial image. Keep in mind that our algorithm uses the method gradient descent, so its starting point is critical. If we initialize differently an image, we may converge to a different minimum point, leading to a different inpainting. In the last two lines we present the case where the holes in the images are too small for the network to notice missing data representing the class. In order to complete missing information in the image, the network must first notice that there are missing parts and features that characterize the classification.

4.4 Special Cases

In this section we present and discuss some special and interesting experiments involving our inpainting algorithm. These experiments do not involve changes in the network or the algorithm.

4.4.1 Inpainting full Images

A special case that we tackled is in which the whole picture is defined as missing and we want to complete the lost information in relation to a certain label. The purpose of this experiment is to present how the network envisions each label.

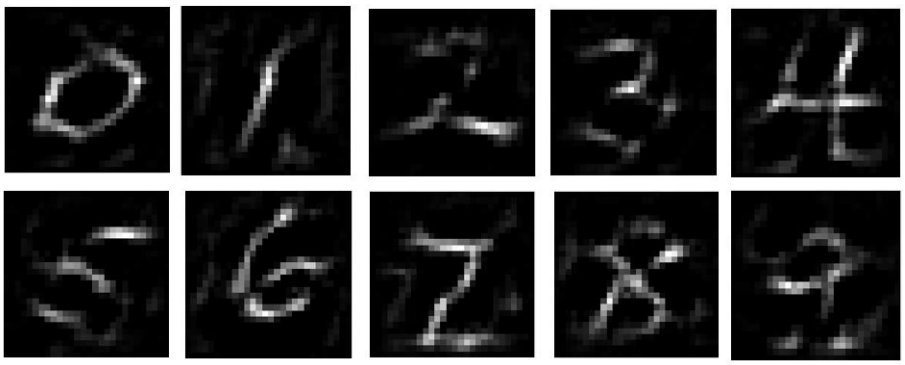

Figure 8: This figure demonstrates the hallucinations of each of the 10 digits (0 to 9).

4.4.2 Illuminating images

We noticed during the experiments with the Yale Extended B database that there are many images taken in the dark, a question then arises: could these images be illuminated with the assistant of our algorithm? We discovered that it is possible by defining the missing parts of the image as all the pixels in the image that are below a specified threshold.

Figure 9: Here we present some of the illuminations for face images. It is possible to distinguish that all the details of the original image still remains after the inpainting. Meaning, all we did is to add information to the dark parts.

4.4.3 Inpainting by all labels

Another experiment we investigated was the case where we tried to inpaint a certain image as all labels and at the end we defined the best inpainting result to be the one which led to the lowest energy function. To our surprise, when we performed this experiment, we received the best inpainting (according to the criterion we defined) for a diffrent label than the original one (as shown in Figure 10).

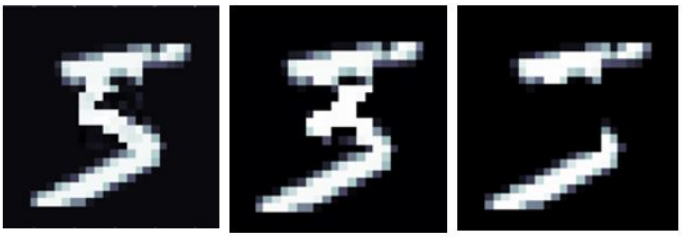

Figure 10: The following figure presents the following experiment. The most-right image is the corrupted image. Notice that the inpainting by label three does seem convincing, so it is not surprising that the best inpainting is for the class three rather than five.

References

[1] Nian Cai, et al., "Blind inpainting using the fully convolutional neural network", Springer 2015

[2] J. Xie, L. Xu, and E. Chen. "Image denoising and inpainting with deep neural networks". In NIPS, 2012.

[3] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. Efros. Context encoders: Feature learning by inpainting. 2016.

[4] Raymond A. Yeh, Chen Chen, T. Yian Lim, A. G. Schwing, M. Hasegawa-Johnson, M. N. Do. "Semantic Image Inpainting with Deep Generative Models". 2017

[5] A. Mahendran and A. Vedaldi. "Visualizing deep convolutional neural networks using natural Pre-Images". In the International Journal of Computer Vision, 2016.

[6] A. Vedaldi and K. Lenc. "MatConvNet Convolutional Neural Networks for MATLAB". Proceedings of the 23rd ACM international conference on Multimedia. 2015.

[7] L. Gatys, A. S. Ecker, and M. Bethge. Texture synthesis using convolutional neural networks. In NIPS, 2015.

[8] C. Guillemot and O. Le Meur, "Image inpainting: Overview and recent advances," Signal Processing Magazine, IEEE, vol. 31, no. 1, pp. 127-144, 2014.

[9] D. Tschumperle, "Fast anisotropic smoothing of multi-valued images using curvature-preserving pde's," International Journal of Computer Vision, vol. 68, no. 1, pp. 65-82, 2006.

[10] A. Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," IEEE Transactions on Image Processing, vol. 13, no. 9, pp. 1200–1212, 2004.

[11] Z. Xu and J. Sun, "Image inpainting by patch propagation using patch sparsity," IEEE Transactions on Image Processing, vol. 19, no. 5, pp. 1153-1165, 2010.

[12] O. G. Guleryuz, "Nonlinear approximation based image recovery using adaptive sparse reconstructions and iterated denoising – part II: Adaptive algorithms," IEEE Transactions on Image Processing, vol. 15, no. 3, pp. 555–571, 2006.

[13] G. Peyre, "Manifold models for signals and images," Computer Vision and Image Understanding, vol. 113, no. 2, pp. 249-260, 2009.

[14] LeCun, Y., Bottou, L., Bengio, Y. and Haffner, P., 1998. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), pp.2278-2324.