Reconstructing Obfuscated Human Faces

Jacob Conrad Trinidad
Stanford University
j3nidad@stanford.edu

Abstract

This paper examines the usage of convolutional neural networks to reconstructed obfuscated images of human faces. Prior work has been done in the field of super-resolution, but this paper will explore the gap using highly obfuscated faces. The input to our algorithm is an obfuscated image of a person's face. We use a convolutional neural network to output a reconstructed image of the person's face. This is be applied to the Labeled Faces in the Wild data set using two types of obfuscation techniques on faces: pixelation and Gaussian blurs. Results will be measured using the PSNR and SSIM metrics. We explore the effect of loss functions of the model by examining pixel loss and perceptual loss functions and showing how perceptual loss is preferable to pixel loss. We also explore the capability of the model to reconstruct faces from highly obfuscated images. Our experiments show that this model can produce recognizable faces from seemingly unrecognizable inputs.

1. Introduction

In photographic media, faces are often obfuscated to protect the identity of those pictured. This obfuscation process is done by removing details from the photograph in order to reduce the identifying facial features. However, these obfuscated images may still contain enough details such that the original features may be able to be recovered. An example of an obfuscated face can be seen in Figure 1.

Two common methods of obfuscation are pixelation and blurring. These methods are often found in image editing software and are used to obfuscate faces. In addition, it is often just the face that is obfuscated because it has the most identifying features, with the rest of the image not obfuscated and in high quality. This paper seeks to test these two methods and examine how well they can prevent reconstructing face of the original image.

The approach we consider is to use convolutional neural networks. The input to the model is an obfuscated image of a face. We then use a CNN to output an image that is a reconstruction of the obfuscated input. This output image will be the same dimensions as the input image. We will examine the effects of using pixel loss as well as using perceptual loss on the model.

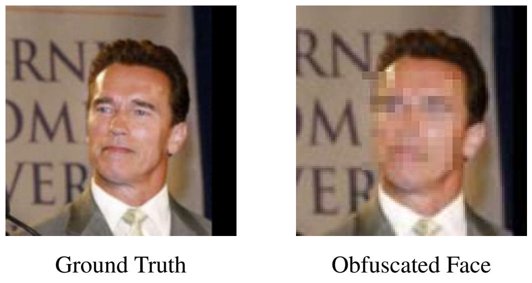


Figure 1: Example of a ground truth image of a person and an obfuscated image.

This paper examines the effects of combining the two loss functions in an attempt to gain the benefits from both loss functions. Furthermore, this paper explores the model's capabilities by using images with high levels of obfuscation and evaluating its performance on reconstructing the original image.

This task is related to super resolution tasks, as we hope to increase the quality from a low quality obfuscated input image to a the high quality image of a face.

2. Related Work

Similar work has been done in the past focusing on obfuscated images and super resolution. Lander *et al*. demonstrates that subjects can still recognize the faces of famous people in static images despite being blurred [14]. Higher accuracy can be achieved by showing the obfuscated faces in motion. Lander *et al*. also shows that obscured faces using methods such as pixelation can still be recognized [13]. Gross *et al*. also shows that privacy is not protected when using low pixelation levels as obfuscated faces can still be recognized and their image quality enhanced [7].

McPherson *et al*. examines the topic of classifying images that underwent various forms of obfuscation highlighting issues with obfuscation [15]. In this paper, they show

how neural networks can classify images that underwent obfuscation processes such as pixelation and blurring by examining different datasets such as MNIST, CIFAR-10, AT&T, and FaceScrub. They demonstrate that neural networks can be used to accurately classify images despite obfuscation processes since these images, while unrecognizable to humans, still contain enough information to enable appropriate classification.

There are four main approaches to the super-resolution task as shown by Yang *et al.*: prediction models, edge based models, image statistical models, and patch based models [22]. Prediction models use interpolation based methods to generate smoother regions in high resolution images. Edge based models learn priors to generate sharp images, but generally lack texture. Image statistical models use image properties to predict the details in higher resolution images. Lastly, patch based models learn mapping functions from smaller lower resolution patches to their corresponding high resolution version.

However, super-resolution may be a task that is suited well for CNNs, which inherently encode mappings from the lower resolution images to higher resolution images. Dong *et al.* apply deep CNNs are able to achieve improvements in super-resolution using a three layer network [6]. Kim *et al.* are further able to improve quality by creating deeper networks with residual blocks [10].

Dahl *et al.* use a probabilistic pixel recursive super-resolution model to upscale an 8x8 image of a face to 32x32 with results that can fool human testers [5]. This work was based off of PixelRNNs and PixelCNNs. PixelRNNs can sequentially predict pixels in an image, through methods such as modeling the distribution and dependencies of pixels in an image [19]. PixelCNNs work similarly to PixelRNNs by modeling it using convolutional neural networks [20].

Of note, Johnson *et al.* examine the task of super-resolution in the context of using a different model loss [9]. In this paper, they explore the perceptual loss function, which focuses on differences between high level image features to improve image transformation tasks. They explore the task of single image super resolution as well as style transfer. In the case of super resolution, they demonstrate that perceptual loss can lead to better reconstructions as it focuses on the fine details and edges in the image.

For super-resolution tasks, common metrics include peak signal to noise ratio (PSNR) and structural similarity (SSIM) [18]. Structural similarity was developed by Wang *et al.* to measure image similarity [21]. However, PSNR and SSIM don't correlate well with human perception of quality. Kundu *et al.* and Sheikh *et al.* both show that PSNR is a poor metric often outperformed by other visual quality metrics [17][12].

3. Methods

3.1. Architecture

The model used to achieve the task of reconstructing obfuscated human faces is an convolutional neural network with an input and output shape of 112x112x3. The model initially subtracts the mean training image and divides each channel by 255.

This is sent through 4 convolutional layers. The first layer consists of 64 9x9 filters. The following three layers consists of 64, 128, and 256 4x4 filters respectively with stride 2. Then we send it through 3 residual blocks. Each block is 2 convolutional layers with 256 3x3 filters. Then it goes through 4 transpose convolution layers. The first three layers have 256, 128, and 64 4x4 filters respectively with stride 2. The last convolutional layer has 3 9x9 filters. All convolution and transpose convolution layers are followed by batch normalization and ReLU activation, except the last layer which is followed by a tanh activation and then scaled to the range [0,255]. The network uses an Adam optimizer to update parameters [11].

3.2. Pixel Loss

For our model, we implemented two types of loss functions. The first loss function we developed was using a squared normalized Euclidean distance between the pixels of the network output ($\hat{y}$) and the ground truth (y). Since the output image of the network is 112x112x3, the equation for this pixel loss is:

$$\ell_{pixel} = \frac{1}{112x112x3}||y - \hat{y}||_2^2$$

3.3. Perceptual Loss

The second loss function we implemented was using a perceptual loss function. We use a loss network to capture the differences in content features in the images. The loss network is pretrained image classification network with its parameters fixed. When using perceptual loss, we feed the loss network both the target ground truth image as well as the network output and then take the feature representation of the images at a certain layer in the network [9]. If the layer is of shape $HxWxC$ and $\phi(x)$ is the feature representation of image x, then the perceptual loss is the squared normalized Euclidean distance between the network output ($\hat{y}$) and the ground truth (y):

$$\ell_{perceptual} = \frac{1}{HWC}||\phi(y) - \phi(\hat{y})||_2^2$$

For our model, we followed the perceptual loss for super-resolution models illustrated in Johnson *et al.* [9]. We used the VGG-16 network pretrained on ImageNet [1]. We then

[1]http://download.tensorflow.org/models/vgg_16_2016_08_28.tar.gz

used the relu2_2 layer from this loss network as the feature representations. This feature representation is of size 56x56x128. We attempt to minimize this loss by backpropagating the differences on the parameters in our network. Note that since the output of our network is 112x112x3 and the VGG network expects an input of 224x224, we reconstruct out own VGG network only up to the stated relu2_2 layer to prevent errors from rising during graph construction.

By using these two types of losses, we can compare examining low level feature representations to high level feature representations. Pixel loss captures low level features by literally minimizing the pixel differences. Perceptual loss captures high level features by minimizing the high level features captured by the VGG network.

4. Dataset

We use the Labeled Faces in the Wild data set [8]. This dataset contains 13,233 images collected from the web. There are a total of 5,749 people pictured in this dataset, with 1,680 people having two or more pictures in this dataset.

Every image in this dataset is 250x250 jpeg image. A benefit of using this dataset is that the images are processed such that faces are detected and centered using the openCV implementation of the Voila-Jones face detector [8]. The images were cropped and enlarged by a factor of 2.2 so the head is fully captured in the image and are at a near uniform size.

We perform some preprocessing on this dataset for this paper. For each image, we first cropping the image from 250x250 to 112x112, maintaining the center of the image. This is the facial region we will focus on for obfuscation. The purpose of this is so we can focus on the reconstruction of the face and ignore the majority of the backgrounds in the images.

Then for each image, we obfuscate it. For each obfuscation method, we copy the original image then run the given obfuscation method.

The first method is pixelation, also known as mosaicing. This is the process of dividing up the area of interest into nxn squares. For each square, the average pixel color is calculated and then every pixel in that square is set to that color, thereby reducing the amount of detail in the are by n^2. A parameter of this obfuscation method is n. As n increases, the higher the obfuscation level and the less detail there is in the original image. For the purposes of the paper, we will use $n = 8$ for most tests as well as try $n = 12$ and $n = 16$ when stress testing the model.

The second method is using Gaussian blurs. In this process, an image is blurred by convolving the area of interest with two one-dimensional Gaussian, resulting in a less blocky version of the original image. A parameter of this obfuscation method is σ which affects the Gaussian kernel. As σ increases, the higher the obfuscation level and the less detail there is in the original image. For the purposes of the paper, we will use $\sigma = 4$ for most tests as well as try $\sigma = 6$ and $\sigma = 8$ when stress testing the model.

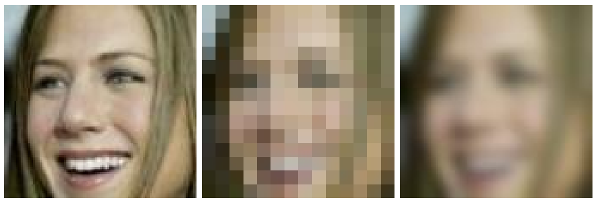

(a) Ground Truth (b) Pixelated Face (c) Blurred Face

Figure 2: Example of processed images from the Labeled Faces in the Wild dataset. **(a)** Illustrates the cropped 112x112 instance of a photo from the data set. **(b)** Illustrates image (a) obfuscated using pixelation with $n = 8$. **(c)** Illustrates image (a) obfuscated using Gaussian blurring with $\sigma = 4$.

This data set was randomly split into 8000 training samples, 2646 validation samples, and 2587 testing samples. An example of this data set can be found in Figure 2.

5. Experiments[2]

For our experiments, we used a batch size of 8 for 15 epochs, which is 15,000 iterations. These values were chosen because initial runs of the model showed that a batch size of 8 maximized PSNR and SSIM metrics and we chose 15 epochs because the PSNR and SSIM metric flattened at around this number of iterations.

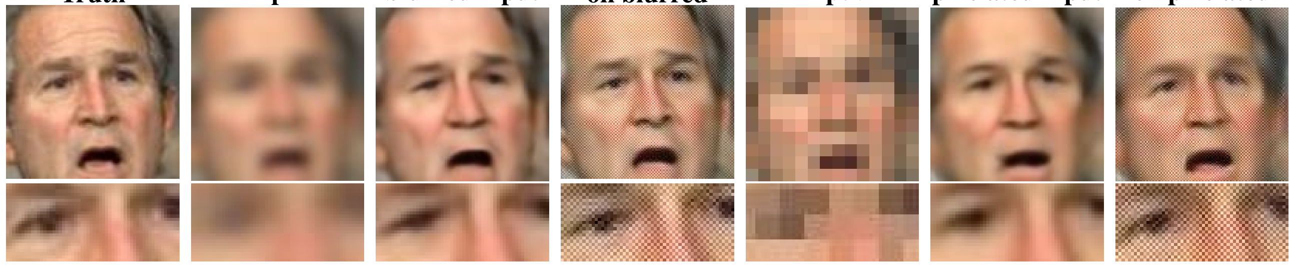
Since we are using the Adam Optimizer in our model, we used a learning rate of $1.0x10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 1.0x10^{-8}$. We used these parameters because we wanted to fix these parameters in order to focus on the effects of different loss functions as well as different levels of obfuscation on the input. These values were chosen because they are generally good initial values as they are the default values in TensorFlow, Adam, and PyTorch [2][3][4].

5.1. Evaluation

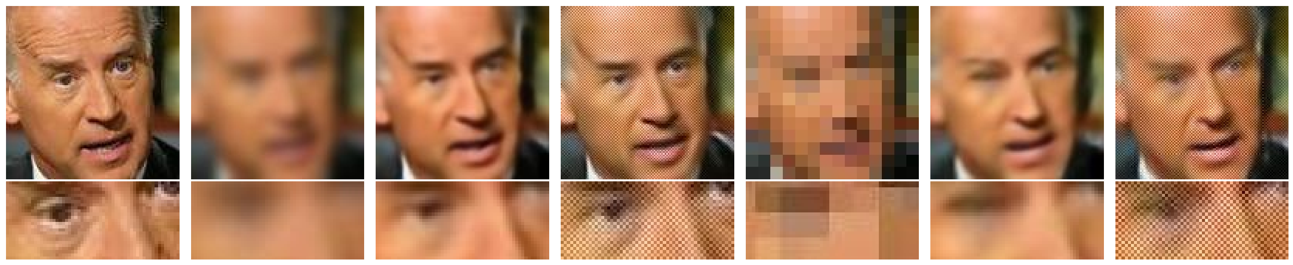
We evaluate the results of the model using primary two metrics: PSNR and SSIM. Recall two metrics are traditional metrics to evaluate super resolution tasks [21][9] [18]. However, since there is a low correlation between these metrics and human perception, we still must rely on intuition and the overall aesthetics of the output [17][12].

PSNR is peak signal to noise ratio, which measures the quality of a reconstruction in terms of absolute errors. The

[2]Code was based on CS231N Assignment 2 and Olivier Moindrot's TensorFlow Example [1][16]



	Ground Truth	Blurred Input	ℓ_{pixel} on blurred input	$\ell_{perceptural}$ on blurred	Pixelated Input	ℓ_{pixel} on pixelated input	$\ell_{perceptual}$ on pixelated
This Image		24.33 / 0.515	29.33 / 0.749	20.72 / 0.332	22.85 / 0.367	27.13 / 0.638	18.61 / 0.234
Training Set		23.74 / 0.471	26.56 / 0.700	21.21 / 0.355	22.36 / 0.326	23.94 / 0.600	19.01 / 0.250



This Image		22.27 / 0.407	25.68 / 0.658	19.86 / 0.337	21.42 / 0.286	23.52 / 0.538	17.77 / 0.234
Validation Set		22.33 / 0.326	25.96 / 0.696	20.82 / 0.338	22.33 / 0.326	23.52 / 0.585	18.55 / 0.227

Figure 3: Results of Pixel Loss vs Perceptual Loss Experiment. A training set example is on the top two and a validations set example is on the bottom. The PSNR / SSIM for the given example and for the dataset are reported. Blurred inputs use $\sigma = 4$ and pixelated inputs use $n = 8$

higher the PSNR the better the reconstruction, generally. In the case of this task, the output of the model is the reconstruction.

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.

We expect our results to ideally match the results seen in Johnson *et al.* as we drew initial from their model architecture. They received a PSNR / SSIM average score of 25.91 / 0.6680 using pixel loss and 24.95 / 0.6317 using perceptual loss on the BSD100 data set [9]. It is important to note that our data set uses lower quality images as well as many more training, validation, and testing images. In addition, our data set focuses only on the human face while the BSD100 data set featured a wide variety of subjects and Johnson *et al.* performs post processing when using perceptual loss. As a result, it is likely for the results to end up significantly more different than expected.

5.2. Pixel Loss vs Perceptual Loss Experiment

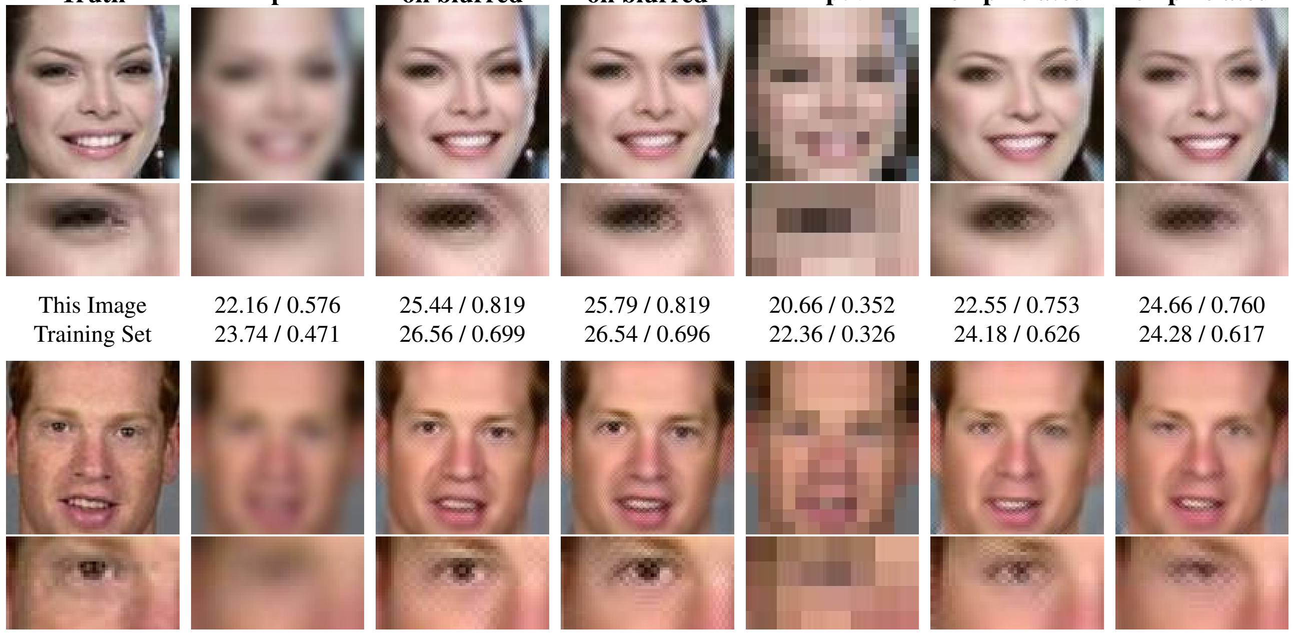
The first experiment was testing pixel loss versus perceptual loss. We evaluated the model architecture using four experiments which compared the results of pixel loss vs perceptual loss on either pixelated input ($n = 8$) or blurred input ($\sigma = 4$) as training data. Figure 3 shows the results of this experiment.

The PSNR / SSIM metrics show that using pixel loss performs better than using perceptual loss, but the examples show that this does not necessarily translate to better images. This outcome is expected as these metrics measure low level differences. In particular, PSNR is dependant directly on the squared euclidean distance between the pixel differences, which is what pixel loss minimizes.

A byproduct of using perceptual loss is the checkerboard like pattern on the output image [9]. This effect helps to cause the drop in PSNR / SSIM metrics for the perceptual loss models.

The results on the training set are only slightly better than the results on the validation set. This suggests that the model can perform well on faces it has not seen to the extent it is able to perform on the training set.

The models trained on perceptual loss generate more details and sharper images despite the obfuscation type. These details can be more clearly seen seen in the whites of the eyes and wrinkles in the zoomed eye regions. In the images generated from perceptual loss, the eyes of whites can be seen much more clearly than those in the pixel loss, which



Ground Truth	Blurred Input	ℓ_{pp} $\rho = 1.0$ **on blurred**	ℓ_{pp} $\rho = 0.5$ **on blurred**	Pixelated Input	ℓ_{pp} $\rho = 1.0$ **on pixelated**	ℓ_{pp} $\rho = 0.5$ **on pixelated**
This Image	22.16 / 0.576	25.44 / 0.819	25.79 / 0.819	20.66 / 0.352	22.55 / 0.753	24.66 / 0.760
Training Set	23.74 / 0.471	26.56 / 0.699	26.54 / 0.696	22.36 / 0.326	24.18 / 0.626	24.28 / 0.617
This Image	24.90 / 0.495	25.13 / 0.721	26.02 / 0.716	23.58 / 0.327	23.06 / 0.649	23.84 / 0.653
Validation Set	22.33 / 0.326	25.59 / 0.683	26.34 / 0.673	22.33 / 0.326	23.18 / 0.580	23.28 / 0.588

Figure 4: Results of the Pixel Loss with Perceptual Loss Experiment. A training set is on the top and a validations set is on the bottom. The PSNR / SSIM for the given example and for the dataset are reported. Blurred inputs use $\sigma = 4$ and pixelated inputs use $n = 8$

appear to generally be black blobs. Furthermore, the facial features are more defined and less soft when using perceptual loss.

Irregardless of loss function, a weakness of the model is its inability to add small facial details such a wrinkles. This effect is demonstrated in the foreheads of the faces, where there are often wrinkles and lines in the ground truth image, but in the model output they are generally smooth. These details are lost in the obfuscation process and are very different for each face. As a result the model cannot determine if wrinkles should exist and if so where. These features are unlike other standard facial features like the eyes, nose and mouth which are located in generally the same positions as well as have a general shape.

In regards to obfuscation, we can see that the model is able to detect edges better when given the blurred obfuscation method in contrst to the pixelated method. Blurring is able to reduce detail but still maintains the relative position of various facial features, allowing for a more accurate reconstruction in comparison to pixelated obfuscation because a lot of specific location detail is lost within each pixelated block.

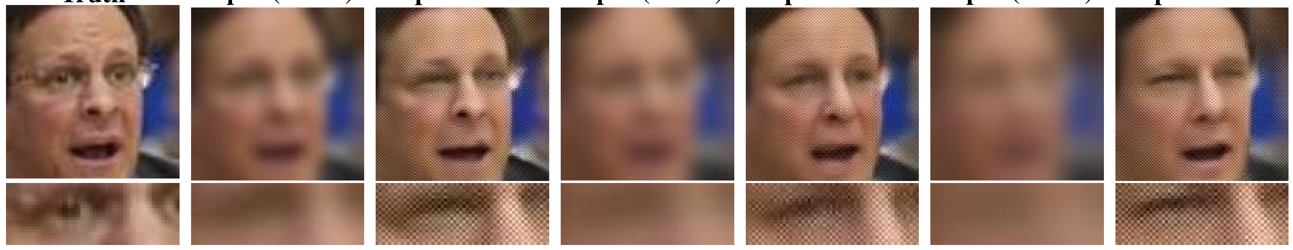
5.3. Pixel Loss with Perceptual Loss Experiment

The second experiment was testing the potential to combine pixel loss and perceptual loss into a single loss function. This is in an attempt to gain the benefits of both loss functions, such that we can obtain the structure and detail of perceptual loss while obtaining pixel level accuracy and removing the checkerboard like nature from the image. Let's defined the combined loss as

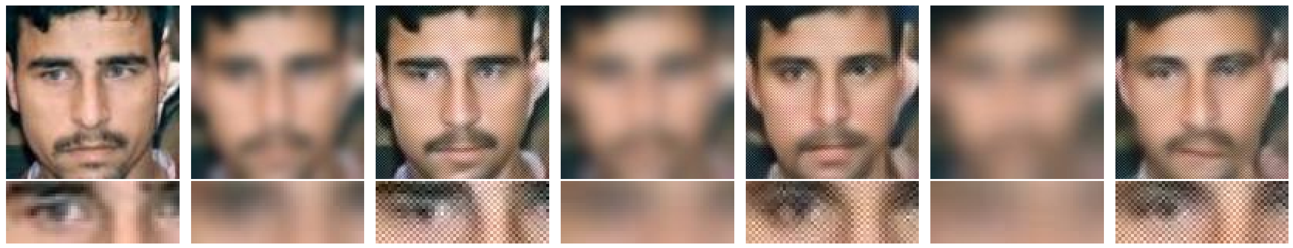
$$\ell_{pp} = \rho \ell_{pixel} + \ell_{perceptual}$$

where ρ is a parameter tweaking the weight between pixel loss and perceptual loss. For this experiment, we ran four models, which compared the effects of using ℓ_{pp} with $\rho = 1$ and $\rho = 0.5$ on both blurred input and pixelated input. When $\rho = 1$, the two losses are weighted equally. When $\rho = 1$, the perceptual loss is weighted twice as much as the pixel loss. We use these two values in our experiment to see if there are potential benefits to combining loss as well as if the value of ρ affects the results. Results for this experiment can be seen in Figure 4

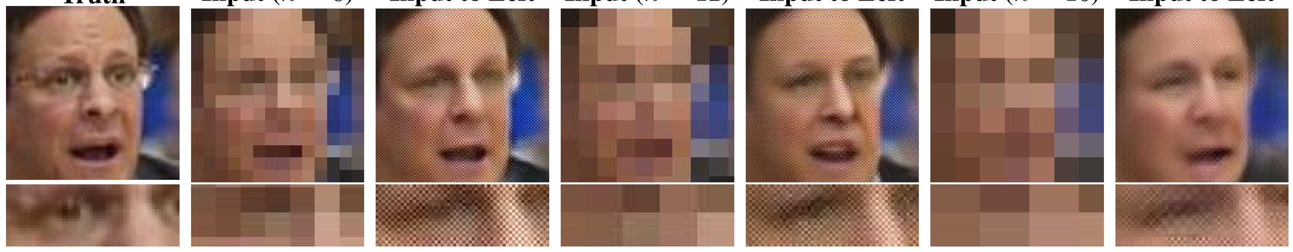
From these results, we can see that combining the losses improves the PSNR / SSIM metrics in comparison to the



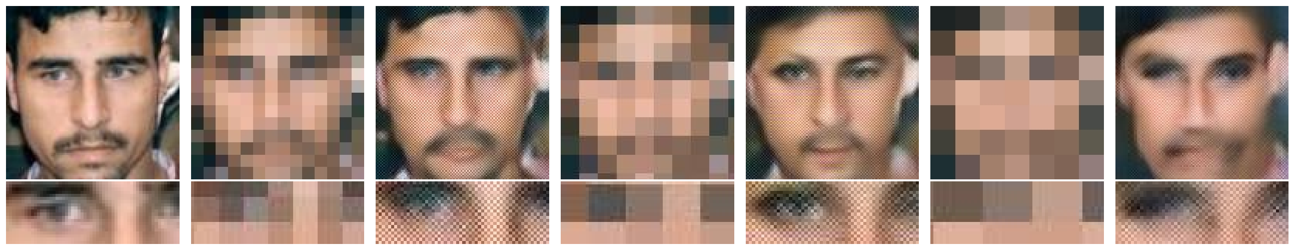
Ground Truth	Blurred Input ($\sigma = 4$)	Output on Input to Left	Blurred Input ($\sigma = 6$)	Output on Input to Left	Blurred Input ($\sigma = 8$)	Output on Input to Left
This Image	25.28 / 0.525	21.50 / 0.323	22.71 / 0.345	23.73 / 0.365	21.24 / 0.239	20.78 / 0.245
Training Set	23.74 / 0.471	21.21 / 0.355	21.67 / 0.322	22.27 / 0.356	20.39 / 0.236	19.89 / 0.243



This Image	20.81 / 0.500	20.73 / 0.458	18.85 / 0.336	20.71 / 0.432	17.66 / 0.262	18.38 / 0.271
Validation Set	22.33 / 0.326	20.82 / 0.338	21.66 / 0.323	21.48 / 0.332	20.37 / 0.237	19.43 / 0.223



Ground Truth	Pixelated Input ($n = 8$)	Output on Input to Left	Pixelated Input ($n = 12$)	Output on Input to Left	Pixelated Input ($n = 16$)	Output on Input to Left
This Image	23.77 / 0.346	19.04 / 0.234	21.57 / 0.186	21.43 / 0.281	20.27 / 0.113	22.27 / 0.352
Training Set	22.36 / 0.326	19.01 / 0.250	20.64 / 0.179	20.54 / 0.274	19.48 / 0.106	20.30 / 0.294



This Image	19.47 / 0.340	17.68 / 0.275	17.82 / 0.193	17.99 / 0.283	16.98 / 0.120	17.46 / 0.243
Validation Set	22.33 / 0.326	18.55 / 0.227	20.62 / 0.178	19.59 / 0.232	19.47 / 0.106	20.08 / 0.264

Figure 5: Results of High Obfuscation Level Experiment. The PSNR / SSIM for the given example and for the dataset are reported. The first set two rows show the model on varying levels of blurring on the training set. The second set two rows show the model on varying levels of blurring on the validation set. The third set of two rows show the model on varying levels of pixelation on the training set. The last set of two rows show the model on varying levels of pixelation on the validation set.

given input, whereas they worsen when using only perceptual loss seen in Figure 3. In addition, when comparing the metrics on the training and validation set on pixel loss from Figure 3 to those in Figure 4, using combined loss maintains or slightly worsens the metrics.

However, there is an issue with using this combined loss function. The checkerboard pattern is still present around the edges of the image. This effect can in particular be seen around the eye. Since there are many edges along the eye, the checkerboard pattern appears, but on the cheeks and forehead were there is minimal edges, there appears to be no checkerboard pattern.

This checkerboard pattern appears to illustrate where the pixel loss is taking more effect and where the perceptual loss is taking more effect. The pixel loss is taking effect primarily on the smooth parts of the image where there is less detail and structure whereas the perceptual loss is taking effect around the complex facial features of the eyes, nose and mouth. This explains why there is more of this checkerboard pattern present when $\rho = 0.5$ than when $\rho = 1.0$, since perceptual loss is stronger when $\rho = 0.5$.

5.4. High Obfuscation Levels Experiment

The third experiment was to test the model architecture on high obfuscation levels. We ran six experiments, which compared the results of the original model architecture on using pixelation with parameters $n = 8, 12, 16$ and on using blurring with parameters $\sigma = 4, 6, 8$. We only used perceptual loss for this experiment. Results for this experiment can be seen in Figure 5.

The results of this experiment show that the model can still generate faces despite high obfuscation levels, with as expected slightly less accuracy. The images seen in the first three sets of rows show that the image output looks very similar to the ground truth. Even higher levels of obfuscation are still able to capture detail such as the whites of the eyes and the direction the eyes are looking irregardless of obfuscation level as seen in the validation example with blurred inputs.

These results also show again the loss in detail for small facial features such as wrinkles. This can be noted in the training example, such that as the obfuscation level grows, the resulting face in the image output looks like a younger version of the subject because these age identifying features are gone. This also shows a weakness in the model with glasses as seen in the training example. Glasses are difficult for the model to generate because of their initial small edges in the ground truth and then disappearance in the images with higher levels of obfuscation.

The validation set example illustrates again the weakness of the model with pixelation. Generally, the higher the pixelation parameter, the less information the image contains, and consequently it is much more difficult to accurately model how facial features will look given this small set of data.

In particular, this can be seen in the facial features of the eyes, nose and mouth. In the more obfuscated input images, the exact location and shape of the eyes are lost. As a result, the model does its best to generate an eye where it believes is an eye, but the details of the eye such as its shape, where the whites are, where the eyes are looking, are lost.

This causes the odd appearances of the eye when $n = 12$ and especially when $n = 16$. In the former, we can see the right eye look forward despite the truth that he's looking to the right. Since most eyes will be looking forward, the model assumed so would these eyes. In the latter, the lack of information in the pixels significantly hurt the detail generated by the network.

The mouth is another instance of this, where the image loses detail on the location of the mouth and the mustache such that they merge on the left side of the image when $n = 16$. These type of results are expected, since for any given low resolution obfuscated image, there are many possible high resolutions that could have generated it.

Note that even at high obfuscation levels, blurring still allows the model to construct accurate faces despite a slight loss in detail.

These results further showcase why PSNR / SSIM metrics are poor metrics of human perception. As we can see in the validation set examples given pixelated inputs, the PSNR metric is almost the same between the highly obfuscated inputs and the output of the network, despite the output of the network looking more pleasing and like a face than the input. The SSIM metrics also sometimes worsen in comparison to the input, which can particularly be seen in the comparison between the second and third column of images.

6. Conclusion

Our model can reconstruct an obfuscated human face. Our experiments show perceptual loss result in more asthetically pleasing faces than pixel loss. We also demonstrate how pixel loss and perceptual loss can be combined to smooth out the checkerboard pattern generated from perceptual loss in regions with fewer edges in the face. Lastly, we illustrate the ability of the model to generate faces from highly obfuscated faces.

Future work can be done in optimizing the hyperparameters of the model, in particular the learning rates as well as well as architecture details such as depth and their effects on the network output. This is when the validation and testing set can be used to show the true potential of the model. In addition, work can be done using more advanced metrics of detecting image similarity rather than PSNR / SSIM.

References

[1] Cs231n convolutional neural networks for visual recognition - assignment 2, 2017.

[2] Optimizers - keras, 2017.

[3] pytorch/adam.py, 2017.

[4] tf.train.adamoptimizer, 2017.

[5] R. Dahl, M. Norouzi, and J. Shlens. Pixel recursive super resolution. *CoRR*, abs/1702.00783, 2017.

[6] C. Dong, C. C. Loy, K. He, and X. Tang. Image super-resolution using deep convolutional networks. *CoRR*, abs/1501.00092, 2015.

[7] R. Gross, L. Sweeney, F. de la Torre, and S. Baker. Model-based face de-identification. In *2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)*, pages 161–161, June 2006.

[8] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[9] J. Johnson, A. Alahi, and F. Li. Perceptual losses for real-time style transfer and super-resolution. *CoRR*, abs/1603.08155, 2016.

[10] J. Kim, J. K. Lee, and K. M. Lee. Accurate image super-resolution using very deep convolutional networks. *CoRR*, abs/1511.04587, 2015.

[11] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[12] D. Kundu and B. L. Evans. Full-reference visual quality assessment for synthetic images: A subjective study. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 2374–2378, Sept 2015.

[13] K. Lander, V. Bruce, and H. Hill. Evaluating the effectiveness of pixelation and blurring on masking the identity of familiar faces. *Applied Cognitive Psychology*, 15(1):101–116, 2001.

[14] K. Lander and L. Chuang. Why are moving faces easier to recognize? *Visual Cognition*, 12(3):429–442, 2005.

[15] R. McPherson, R. Shokri, and V. Shmatikov. Defeating image obfuscation with deep learning. *CoRR*, abs/1609.00408, 2016.

[16] O. Moindrot. Example tensorflow script for fine-tuning a vgg model (uses tf.contrib.data), 2017.

[17] H. R. Sheikh, M. F. Sabir, and A. C. Bovik. A statistical evaluation of recent full reference image quality assessment algorithms. *IEEE Transactions on Image Processing*, 15(11):3440–3451, Nov 2006.

[18] R. Timofte, V. De, and L. V. Gool. Anchored neighborhood regression for fast example-based super-resolution. In *2013 IEEE International Conference on Computer Vision*, pages 1920–1927, Dec 2013.

[19] A. van den Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. *CoRR*, abs/1601.06759, 2016.

[20] A. van den Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu. Conditional image generation with pixelcnn decoders. *CoRR*, abs/1606.05328, 2016.

[21] Z. 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, April 2004.

[22] C.-Y. Yang, C. Ma, and M.-H. Yang. Single-image super-resolution: A benchmark. In *Proceedings of European Conference on Computer Vision*, 2014.