How Secure are Deep Optical Privacy Preserving Methods?

Edwin Pan*
Stanford Electrical Eng.
Stanford, CA 94305

Anthony Vento*
Stanford Electrical Eng.
Stanford, CA 94305

Claire Zhang*
Stanford Electrical Eng.
Stanford, CA 94305

Abstract

Privacy preservation is a critical issue in computer vision. To ensure privacy, recent works in deep optical encoding schemes apply learned optical blurs at the lens level. Such techniques are considered the gold standard in privacy preservation, since encoded images never produce a digital representation of the sharp image. The physical light coming into the photo-sensor is already obfuscated, yet downstream vision tasks like pose estimation and action recognition can still be performed [1]. In this project, we attempt a more nefarious downstream vision task. Using the information preserved in the optically encoded images, we attempt to generatively reconstruct the obfuscated fine details. To this end, we propose a blind deconvolution model titled **De**convolving **D**eep **O**ptical Encoders with Learned **S**amples (**DeDOS**). Concretely, given a trained image generator $\mathcal{G}_I$ and blur kernel generator $\mathcal{G}_K$, DeDOS takes an input encoded image i_e and optimizes latent vectors (z_i, z_k) to best recreate $\hat{i}_b = \mathcal{G}_I(z_i) * \mathcal{G}_K(z_k)$. The generated image $\hat{i}_s = \mathcal{G}_I(z_i)$ should 'revert' the privacy-preserved image i_e to its sharp forms and recover hidden private attributes. Code is available at `https://github.com/edwin-pan/dedos`.

1 Introduction

The advent of computer vision models for scene understanding has helped incentivize the wider deployment of intelligent optical surveillance systems. As video feeds generally capture far more information than is necessary to understand the components of a scene (i.e. actions, objects, poses, etc), an interesting line of research involves the partitioning of visual information. Namely, how can we separate the visual information such that we retain only what is strictly necessary to perform our desired scene analysis. Work in this area has gone so far as to separate the unnecessary information at the optical level, resulting in imaging pipelines that never digitally represent the sharp image but can still perform common downstream computer vision tasks. While directly tapping into the pipeline does not allow someone to view the scene being imaged, it may still be possible to use the information retained in the encoded image to generatively reconstruct the sharp image.

The primary goal of this project is to learn one such generative model that can recover a visually plausible sharp representation given the optically encoded image. Our method, titled **De**convolving **D**eep **O**ptical Encoders with Learned **S**amples (**DeDOS**) represents our best attempt to break this optical encoding. Deep optical privacy preserving pipelines have the potential to serve in a wide range of sensitive surveillance applications. It is therefore vital that their robustness be thoroughly tested prior to mass adoption.

*Equal contribution, alphabetical by last names.

Preprint. Under review.

2 Related Work

GAN based attacks on deep optical encoding schemes are, as a class of assault, not new. Hinojosa et. al fine tuned an off-the-shelf motion deblurring GAN for this task and found that the deep optics did a decent job at preserving private information (faces, identity, etc) even in the hardest scenarios [1]. However, the argument can be made that more sophisticated GANs may be able to better reconstruct private information. Additionally, Ulyanov et al show that a randomly initialized Neural Network used as a prior can successfully denoise most images [2]. Ramakrishnan et al use a general adversarial network with a global skip connection and a dense architecture to perform blind motion deblurring [3]. Finally, Zhang et al develop a light weight generator network and append a term to the loss to remove grid artifacts of certain images [4].

In Asim et al [5], blind deconvolution is achieved by learning image priors and kernel priors from a given training set of blurry images via GAN networks. Inference is done by searching for an optimal z_i and z_k latent embedding within the learned priors of image and kernel respectively. Search is done via gradient descent (i.e. fixing the weights of the image/kernel generators and optimizing a proxy loss between encoded test image and generated image/kernel).

There have also been attempts to control specific detail in a generated image. StyleGan [6] achieves fine-grained control over generated image details by injecting noise vectors in the generator at each resolution stage. Such an approach could control the reconstruction of fine-grained private attributes.

Combining the latent-vector optimization techniques from [5] with the fine-grained feature control from [6], our approach (DeDOS) present a much more sophisticated reconstruction attack to the given Zernike polynomial based deep optical encoding scheme in [1].

3 Approach

Our approach has several key components. In this section, we describe the dataset used to train and evaluate our models, the post-processing optimization scheme (latent vector optimization), and the various generator architectures we test. The choice of blur kernel (i.e. Zernike based parameterization) and DeblurGANv2 was to allow our results to be comparable to those originally reported in [1]. Additionally, the original authors that proposed latent vector optimization [5] used PG-GAN [7] as their image generator. We include this as a candidate image generator architecture.

3.1 Dataset

For this project, we use a custom set of 90K images sampled from the COCO object detection dataset [8]. Each image has been fed through the deep optical encoding scheme found in [1], resulting in 90K pairs of (encoded, sharp) images. Example images are shown in Figure 1. Unfortunately, the encoding method [1] is presently patent pending and legal barriers prevent us from performing analysis on other more structured datasets (e.g. CelebA [9]).

Figure 1: Example samples of encoded and sharp image pairs from the custom dataset.

3.2 Latent Vector Optimization

Our approach follows the optimization scheme described in [5], with generator architectures detailed in the sections below. We have encoded images $i_e \in \mathbb{R}^{256\times256}$ derived from an unknown

transformation kernel $k \in \mathbb{R}^{n \times n}$ from class $\mathcal{K}$ applied on sharp images $i_s \in \mathbb{R}^{256 \times 256}$ from class $\mathcal{I}$. We denote generators $\mathcal{G}_\mathcal{I}$ and $\mathcal{G}_\mathcal{K}$ as mappings $\mathcal{G}_\mathcal{I} : \mathbb{R}^l \rightarrow \mathbb{R}^{256 \times 256}$ and $\mathcal{G}_\mathcal{K} : \mathbb{R}^{350} \rightarrow \mathbb{R}^{n \times n}$ respectively, where low-dimensional $z_i \in \mathbb{R}^l$ is a latent vector input for $\mathcal{G}_\mathcal{I}$ and $z_k \in \mathbb{R}^{350}$ are the first 350 coefficients for the Zernike polynomials used in $\mathcal{G}_\mathcal{K}$. We train the generator $\mathcal{G}_\mathcal{I}(z_i)$ to obtain representatives of class $\mathcal{I}$. The kernel generator $\mathcal{G}_\mathcal{K}(z_k)$ is the fixed Zernike polynomial expansion of $\mathcal{K}$. We then freeze the weights of $\mathcal{G}_\mathcal{I}$ and $\mathcal{G}_\mathcal{K}$ to optimize the sampled latent vectors z_i and z_k from which $\mathcal{G}_\mathcal{I}$ and $\mathcal{G}_\mathcal{K}$ generate reconstruction $\hat{i_s}$ and blur kernel $\hat{k}$. We minimize the objective function

$$(\hat{i}_s, \hat{k}) = \text{argmin}\mathcal{L}(z_i, z_k), \quad i \in \text{Range}(\mathcal{G}_\mathcal{I}), k \in \text{Range}(\mathcal{G}_\mathcal{K}) \tag{1}$$

with concrete loss values computed in the latent vector space. This loss is a weighted sum of L2, cosine similarity, and total variation (TV) computed on the sharp reconstructed image.

$$\mathcal{L}(z_i, z_k) = ||i_e - \hat{i}_b||_2 + \lambda(1 - \text{CosSim}(i_e, \hat{i}_b)) + \gamma||\mathcal{G}_\mathcal{I}(z_i)||_{tv} \tag{2}$$

Note that the blurry reconstruction $\hat{i}_b$ is defined as:

$$\hat{i}_b = \mathcal{G}_\mathcal{I}(z_i) * \mathcal{G}_\mathcal{K}(z_k) \tag{3}$$

Algorithm 1 DeDOS Latent Vector Optimization Procedure

procedure LATENTVECTOROPTIMIZER($i_e, \mathcal{G}_\mathcal{I}, \mathcal{G}_\mathcal{K}$)
  Initialize: $z_i^{(0)} \sim \mathcal{N}(0, I)$, $z_k^{(0)} \sim \mathcal{N}(0, I)$
  for t **in** (0,1,$\cdots$,T):
    $z_i^{(t+1)} = z_i^{(t)} - \eta \nabla_{z_i} \mathcal{L}(z_i^{(t)}, z_k^{(t)})$ ▷ Update using gradient w.r.t z_i
    $z_k^{(t+1)} = z_k^{(t)} - \eta \nabla_{z_k} \mathcal{L}(z_i^{(t)}, z_k^{(t)})$ ▷ Update using gradient w.r.t z_k
  end for
  $\hat{i}_s = \mathcal{G}_\mathcal{I}(z_i^{(T)}), \quad \hat{k} = \mathcal{G}_\mathcal{K}(z_k^{(T)})$
  return $\hat{i}_s, \quad \hat{k}$

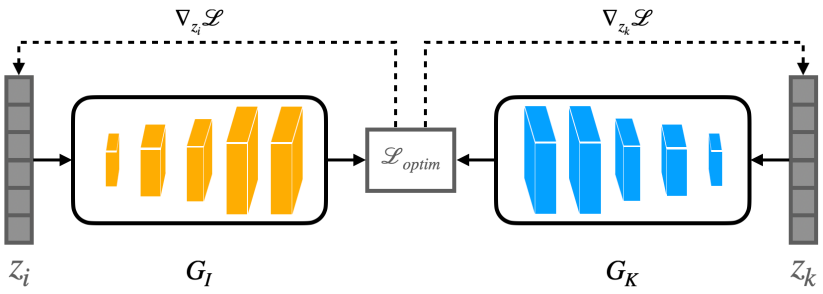

Figure 2: High level diagram of latent vector optimization procedure.

3.3 Generator Architectures & Methodology

In this work, the final DeDOS model uses a custom image generator model (DeblurGANv2++) fine tuned on our custom dataset as $\mathcal{G}_\mathcal{I}$ in Figure 2, along with a deterministic blur kernel generator ($\mathcal{G}_\mathcal{K}$) defined by the first 350 Zernike bases and parameterized by 350 coefficients. What follows is a breakdown of $\mathcal{G}_\mathcal{K}$, along with each of the different image generators $\mathcal{G}_\mathcal{I}$ we evaluated.

3.3.1 Zernike-based Blur Kernel Generator

As mentioned previously, we parameterize the blur kernel generator $\mathcal{G}_\mathcal{K}$ with 350 coefficients $z_k = \alpha$. Each parameter α_i is element-wise multiplied by the corresponding Zernike basis point spread function (PSF) Z_i (Figure 3). By this definition, we can obtain the blur kernel computed at each iteration via

$$k = \sum_{i=0}^{349} \alpha_i Z_i \tag{4}$$

The final blurry image $\hat{i}_b$ is obtained by multiplying the generated optical transfer function (OTF) with the generated sharp image in the frequency domain, then transforming back to the spatial domain.

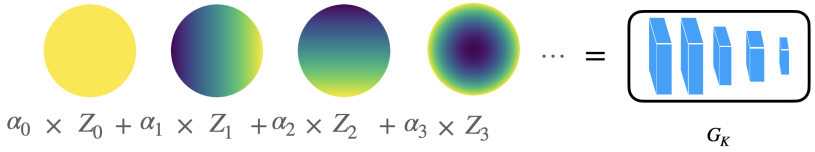

Figure 3: High level diagram of the Zernike-based blur kernel generator described above.

3.3.2 Progressively Growing GAN (PG-GAN)

PG-GAN [7] is a GAN training methodology that progressively grows the generator and discriminator to generate output from a lower resolution to a higher resolution (Figure 4). During a resolution transition, two resolutions are interpolated with weight α linearly increasing from 0 to 1 to fade in the new resolution layer smoothly. Since this GAN architecture was the original model used in [5], we train this model on our custom dataset for baseline analysis.

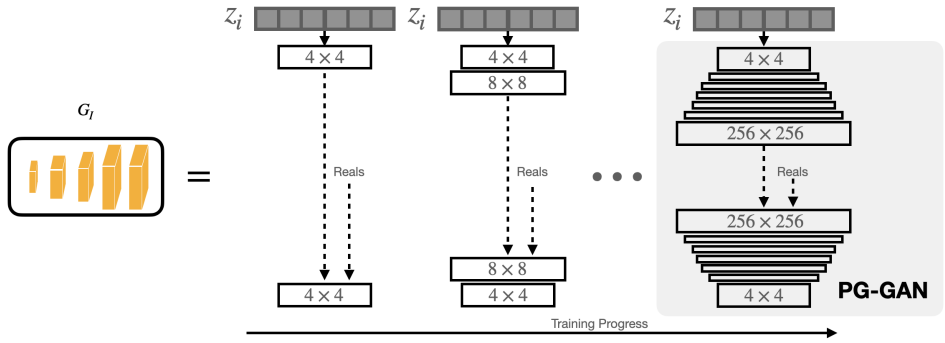

Figure 4: Diagram of PG-GAN training progress & architecture.

3.3.3 DeblurGANv2 & DeblurGANv2++

DeblurGANv2 DeblurGANv2 [10] architecture uses a feature pyramid network (FPN) module containing a bottom-up encoding pathway and a top-down decoding pathway. Feature maps of each encoding stage are extracted and combined to the corresponding decoding stage. Backbone architectures studied by [10] include Inception-ResNet-v2 [11], MobileNet v2 [12], and MobileNet-DSC, the first of which is evaluated in our preliminary result. DeblurGANv2 uses two discriminators that respectively evaluate at a global and a local scale. The loss objective function is $L_G = \alpha L_p + \beta L_X + \gamma L_{adv}$ where L_p is pixel-space loss, L_X is perceptual loss, and L_{adv} contains both discriminators' losses. α, β, and γ are controllable parameters.

DeblurGANv2++ Although capable of generating plausible reconstructions alone, the original DeblurGANv2 model architecture has two problems that make it insufficient for our latent vector optimization scheme.

1. There is no latent vector z_i. The output $\hat{i}_s$ is deterministic given an input encoded image i_e.
2. DeblurGANv2 reconstructed sharp images lack fine details.

To address these issues, we implement and evaluate noise injection into the DeblurGANv2 reconstruction pipeline. Similar to the way StyleGAN [6] injects noise, we add random gaussian noise

arrays at each convolution layer of increasing spatial resolution (Figure 5). Each noise injection layer comes with an additional weight parameter. This weight parameter controls how much influence any specific noise value has on the intermediate reconstructions, and is learned during the training process. The training process and loss terms are unaltered relative to the original DeblurGANv2 model. We dub this noise-injected model DeblurGANv2++. This model has both tune-able latent vectors z_i and superior fine-grained detail reconstruction.

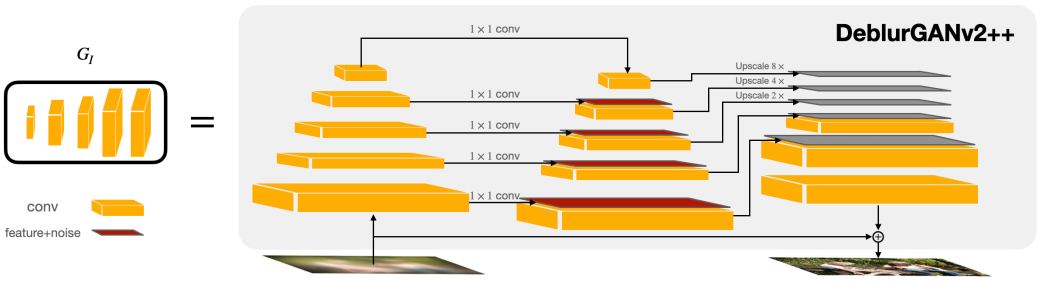

Figure 5: Diagram of proposed DeblurGANv2++ which consists of DeblurGANv2 + StyleGAN-esque noise injection

We note that the injection of noise introduces new content into the image that manifests as the finer-grained details missing in the encoded image i_e [6]. Given a well trained DeblurGANv2++, we can perform a search over all 'missing' details parameterized by z_i in latent vector optimization, and ultimately arrive at the best fine-grained details that explain the observation i_e.

4 Results

Due to the large dataset size, we applied our methodology to a randomly generated 100-sample subset of the held-out test set. We report two different quantitative measures of global image quality.

1. PSNR: peak-signal-to-noise-ratio quantifies a global reconstruction quality with respect to corrupting noise and is measured in decibels (dB). The higher this metric is, the better.
2. SSIM: structural-similarity-index-metric [13] quantifies similarity between two given images using pixel-level statistics. A structural similarity of 1 indicates that the two images are identical. The higher this metric is, the better.

We choose SSIM and PSNR as performance metrics for their general applicability in image quality assessment across various applications. Each model is evaluated on the same 100 image test set.

4.1 Qualitative Results

4.1.1 PG-GAN

Asim et al was able to achieve decent visual reconstructions in their latent vector optimization scheme using PG-GAN as their image generator backbone for the CelebA [9] dataset. We implemented and trained a PG-GAN [7] on the custom dataset with the aim of generating images similar to those found in the COCO dataset. Since the post optimization deblurring scheme assumes that the sharp image can be created through the trained image generator $\mathcal{G}_I$ with learned latent vector z_i, it's imperative that $\mathcal{G}_I$ (PG-GAN in this case) produce quality images. After around two days of training, the model produces images seen in Figure 6 and Figure 7. This PG-GAN was trained with the following {`res:batch_size`} scheduling: {`4:32, 8:32, 16:32, 32:16, 64:16, 128:16`}.

From a visual perspective, it is very unclear what images are being generated. We sort of see resemblance to waves and/or bushes in a number of images, but again, it is not entirely clear. These results indicate that PG-GAN is unable to produce sharp enough images, especially given it intended role as $\mathcal{G}_I$ for the deblurring optimization scheme. This is expected, since COCO is a diverse dataset of in-the-wild images exhibiting very little common structure (compared to faces in CelebA [9]).

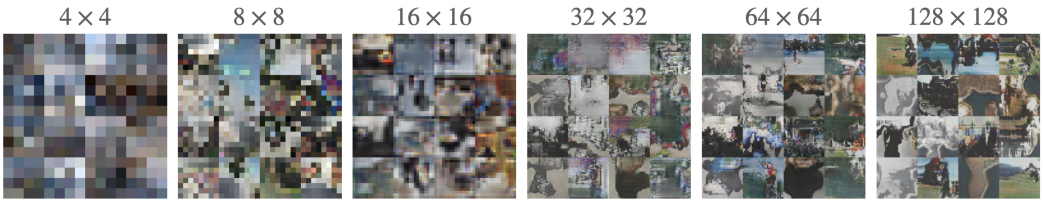

Figure 6: 4x4 Samples from PG-GAN at various resolutions during the training process.

Figure 7: Single image samples from PG-GAN at various resolutions during the training process.

4.1.2 DeblurGANv2, DeblurGANv2++, & DeDOS

Fig 8 shows results from qualitative results for two images for DeblurGANv2, DeblurGANv2++, & DeDOS. The baseline DeblurGANv2 and DeblurGANv2++ were trained for around two days with the same learning rate of 0.0001. We note again that DeDOS is our two stage pipeline which consists of DeblurGANv2++ and the Latent Vector Optimization. For the Latent Vector Optimization, we ran 1000 iterations per image with a learning rate of 0.01.

Depending on what we deem as perceptually plausible, we can generally see an improvement as we progressively improve the model through DeblurGANv2 (baseline), DeblurGANv2++, & DeDOS. To start, in the bus images, all models are able to recover the structure of the bus and some of the hidden finer details (i.e. headlight, license plate, etc). However, the color pallet of the bus is distorted in DeblurGANv2's reconstruction. Both DeblurGANv2++ and DeDOS reproduce sharp predictions that exhibit a color pallet more faithful to the original image.

Figure 8: Example qualitative results of different models

For the image of the people, all models are able to give context to the scene and depict that it is two children talking. Once again we observe that DeblurGANv2++ and DeDOS produce a more accurate color pallet. In the DeblurGANv2++ reconstruction, there are several unsightly artifacts in the background and on the children's faces that reduce perceived image reconstruction quality. DeDOS's reconstruction, although blurrier, significantly reduces the impact of these artifacts are thus produces an (arguably) superior visual reconstruction.

Ultimately, judging visual quality qualitatively is an inherently subjective game and the quality of each reconstruction is best left for the reader to judge.

4.2 Quantitative Results

Table 1 summarizes the PSNR and SSIM values for four sets of models. The original encoded images, Encoded, have the lowest PSNR and SSIM scores. Direct application of baseine model DeblurGANv2 [10] improves the metrics. Our DeblurGANv2++ augmented with noise injection lifts the metrics even more. Lastly, our full model DeDOS, which is further enhanced with latent vector optimization, reaches the highest PSNR score of 17.54 dB and the highest SSIM score of 0.442. From the numerical perspective, DeDOS does achieve the superior performance in the blind deconvolution task.

Model	**PSNR** ↑	**SSIM** ↑
Encoded [1]	12.48	0.353
PG-GAN[2] [7]	–	–
DeblurGANv2 [10]	16.64	0.384
DeblurGANv2++	17.29	0.410
DeDOS (ours)	**17.54**	**0.442**

Table 1: Quantitative results of blind deconvolution methods

Table 2 shows results for different weights on terms in the latent vector optimization loss functions. As with the results in Table 1, all metrics are computed on images sampled from the test set. The term λ weights the cosine similarity term, and the γ term weights the total-variation (TV) prior term (see Eq 2).

λ	γ	**PSNR** ↑	**SSIM** ↑
1	0.001	17.51	0.442
1	0.00075	17.04	0.432
20	0.001	17.50	0.441
5	**0.001**	**17.54**	**0.442**

Table 2: Subset of loss function hyperparameters

We note that there is a correct balance of choosing a high value for the cosine similarity term while choosing a value that is not too high for the TV prior.

4.3 Result Analysis

From Table 1, following the progression of listed models, we observe that each of our proposed improvements increase the PSNR and SSIM scores. From baseline model DeblurGANv2 [10] to DeblueGANv2++, the addition of noise injection potentially may have led to higher scores by allowing the model to leverage the trainable weighted noise to create fine-grained details. From DebluGANv2++ to DeDOS, the addition of latent vector optimization may have improved the metrics by allowing the model to select the latent vector containing the most useful representation for sharp scene reconstruction.

[2]PG-GAN quantitative results omitted because reconstructed visual quality was deemed too poor.

When comparing the qualitative and quantitative results, we observe that higher numerical PSNR and SSIM values do not guarantee more realistic appearance in the reconstructed images. For example, in the bottom row of Figure 7, DeblurGANv2++ produces higher PSNR but results in unusual sharp pattern on the child's face, and DeDOS has the highest values for both SSIM and PSNR but its image looks blurrier.

Therefore, we need to rethink PSNR and SSIM's capability to capture perceptual authenticity, and whether there are more suitable metrics to optimize in order to close the perceptual difference between prediction and ground truth.

5 Conclusion

From the qualitative results, we see that DeDOS is capable of effectively reconstructing some plausible details that are optically hidden in the encoded image. Quantitatively, reconstruction quality in DeDOS generated sharp images is superior to those generated from using DeblurGANv2 or DeblurGANv2++ alone.

5.1 Limitations

In the vision space, numerical measures of image quality often do not correlate with perceived image quality. Quantifying the privacy of an image is arguably an even harder problem. Gauging the implications DeDOS's capabilities largely depends on how "private" is private enough. For instance, we observed that actions/activities being performed in an image are generally recoverable (i.e. baseball, skiing, surfing, reading, etc). If this information were to be considered compromising, then deep optical encoded images are not secure enough. If, however, the desired information resides in finer-grained details (ex: facial features for determining identity, visible lettering to read written documents) then the security of deep optical encoding methods has been maintained. As such, it really depends on each individual to observe the encoded image, observe the reconstruction, then observe the sharp image, and ask themselves at each stage: "what is this image depicting?".

5.2 Future Work

Due to the loose structure within the space of all possible details in natural images, generating these details (i.e. sharp reconstructions) is a hard task. Future work in this area would need to define proxy tasks that target more concrete questions. Specifically, knowing "who", "what, and "where", generally provides enough information to make an educated guess about scene content. To this end, we note the following possible extensions.

- **Who**: Run pose detection on the encoded images (we know this works from [1]), then reconstruct facial features alone using pixels obtained from the bounding box of the face.
- **What**: Augment the training loss of DeblurGANv2++ with a score for object detection consistency between the predicted sharp image and the ground truth sharp image using some existing pre-trained object detection network. This would bias the model towards generating images that preserve object detectability, and provide a more targeted evaluation metric beyond PSNR & SSIM, which are blunt measures of perceptual plausibility at best.
- **Where**: The existing DeDOS model generally does well on this task.

Although this solution is not as clean as simply reconstructing the original sharp image, this decomposed approach to parsing the taxonomy of a scene might yield more privacy concerns than the one-shot task attempted in this work.

6 Acknowledgements

This work was completed as a final project for CS 236: Deep Generative Models. We would like to thank Carlos Hinojosa for his support and for supplying us with the dataset. Additionally, we would like to thank our instructors, Professor Stefano Ermon and Yang Song, for their valuable instruction this quarter. Finally, we would like to thank our project mentor, Mihir Patel, for his insightful remarks.

References

[1] Carlos Hinojosa, Juan Carlos Niebles, and Henry Arguello. Learning privacy-preserving optics for human pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2573–2582, October 2021.

[2] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9446–9454, 2018.

[3] Sainandan Ramakrishnan, Shubham Pachori, Aalok Gangopadhyay, and Shanmuganathan Raman. Deep generative filter for motion deblurring. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 2993–3000, 2017.

[4] Ada Zhen, Robert L Stevenson, et al. Gan based image deblurring using dark channel prior. *Electronic Imaging*, 2019(13):136–1, 2019.

[5] Muhammad Asim, Fahad Shamshad, and Ali Ahmed. Blind image deconvolution using deep generative priors, 2019.

[6] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks, 2019.

[7] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation, 2018.

[8] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2015.

[9] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[10] Orest Kupyn, Tetiana Martyniuk, Junru Wu, and Zhangyang Wang. Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better, 2019.

[11] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 4278–4284. AAAI Press, 2017.

[12] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks, 2019.

[13] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.