Following we provide additional visual and quantitative results. Additionally we explain the attack framework.

A. Attack Framework

Let \( x \) denote the ground-truth image, which is corrupted by a possibly non-linear degradation operator \( \mathcal{A} \), resulting in an observation \( y^{\text{clean}} \), which can be expressed as

\[
y^{\text{clean}} = \mathcal{A}(x).
\]

Let \( \mathcal{G}_\theta \) be a (Transformer-based) neural network parameterized by \( \theta \) trained to recover \( x \) from \( y^{\text{clean}} \). In this work, we are interested in studying the stability of \( \mathcal{G}_\theta \) to adversarial attacks that aim to degrade its performance through visually imperceptible changes to the inputs \([18, 32]\). We evaluate the robustness to attacks using additive perturbations \( \delta \) of the reconstruction error. The loss (MSE loss) to obtain adversarial examples maximizing the deviation of the network output from the ground truth as measured by a loss function \( L \), subject to \( \ell_p \) norm constraints on the perturbation:

\[
\max_\delta L(\mathcal{G}_\theta(y^{\text{clean}} + \delta), x) \quad \text{s.t.} \quad ||\delta||_p \leq \epsilon. \tag{6}
\]

PGD. PGD is an iterative adversarial attack, where each sample is perturbed for a fixed amount of attack iterations (steps) with the intention of maximizing the loss further with each attack step. A single attack step in the PGD attack \([32]\) is given as follows,

\[
y^{\text{adv}t+1} = y^{\text{adv}t} + \alpha \cdot \text{sign} \nabla y^{\text{adv}t} L(\mathcal{G}_\theta(y^{\text{adv}t}), x) \tag{7}
\]

\[
\delta = \phi^\epsilon(y^{\text{adv}t+1} - y^{\text{clean}})
\]

\[
y^{\text{adv}t+1} = \phi^\epsilon(y^{\text{clean}} + \delta)
\]

where the adversarial example \( y^{\text{adv}t+1} \) at step \( t+1 \), is updated using the adversarial example from the previous step \( y^{\text{adv}t} \), \( \nabla \) represents the gradient operation, \( \alpha \) is the step size for the perturbation, \( \phi^\epsilon \) is denoted projection onto the appropriate \( \ell_p \)-norm ball of radius \( \epsilon \), depending on the \( \ell_p \) norm constraints on \( \delta \), and \( \phi^\epsilon \) clips the adversarial example to lie in the valid intensity range of images (between \([0, 1]\)). Prior works evaluating the adversarial robustness of image restoration networks consider \( L \) to be the reconstruction loss (MSE loss) to obtain adversarial examples maximizing the reconstruction error.

CosPGD. Instead of directly utilizing the averaged pixel-wise losses in PGD attack steps, \([1]\) propose to weigh the pixel-wise losses using the cosine similarity between the network output and the ground truth (both scaled by softmax), to reduce the importance of the pixels which already have a large error in the previous iterations, and enable the attack to focus on the pixels with low error. For the task of restoration (a regression task), CosPGD attack steps for an untargeted attack are given as:

\[
x^{\text{adv}t} = \mathcal{G}_\theta(y^{\text{adv}t}) \tag{8}
\]

\[
L_{\text{cos}} = \sum \cos \left( \psi(x^{\text{adv}t}), \psi(x) \right) \odot L(x^{\text{adv}t}, x)
\]

\[
y^{\text{adv}t+1} = y^{\text{adv}t} + \alpha \cdot \text{sign} \nabla y^{\text{adv}t} L_{\text{cos}}
\]

\[
\delta = \phi^\epsilon(y^{\text{adv}t+1} - y^{\text{clean}})
\]

\[
y^{\text{adv}t+1} = \phi^\epsilon(y^{\text{clean}} + \delta),
\]

where \( \psi \) is the softmax function, \( \odot \) denotes point-wise multiplication, and the cosine similarity (cossim) is given by

\[
\cos \left( \frac{\mathbf{u}, \mathbf{v}}{||\mathbf{u}|| \cdot ||\mathbf{v}||} \right)
\]

\([1]\) demonstrate that this approach results in a stronger attack for pixel-wise regression tasks than a PGD attack.

B. Additional Results

We provide sample reconstructed images from all considered networks under adversarial attacks. Figure \([A1]\) shows reconstructed images from GoPro test dataset \([35]\) after the CosPGD attack \([1]\) on the models. Whereas Figure \([A2]\) shows reconstructed images from GoPro test dataset \([35]\) after the PGD attack \([32]\) on the models.

B.1. Intermediate networks

Further, we discuss some additional implementation details pertaining to the Intermediate networks and provide further observations and insights on their performance.

In Table \([A1]\) we report the performance of the Intermediate network and Intermediate + ReLU. Please note, the performance of the Intermediate network on the clean (unperturbed) samples is marginally lower than that reported by \([7]\). As \([7]\) does not provide the code, pre-trained weights, or training configuration for this intermediate step between the Baseline network and NAFNet, our implementation is limited to the best of our understanding.
### Table A1. Comparison of performance of all the considered models with $\alpha=0.01$ and $\epsilon=8.255$.

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Clean</th>
<th>CosPGD</th>
<th>PGD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR</td>
<td>SSIM</td>
<td>PSNR</td>
</tr>
<tr>
<td></td>
<td>5 attack itrs</td>
<td>10 attack itrs</td>
<td>20 attack itrs</td>
</tr>
<tr>
<td>Restormer</td>
<td>31.99</td>
<td>0.9635</td>
<td>11.36</td>
</tr>
<tr>
<td>+ ADV</td>
<td>30.25</td>
<td>0.9453</td>
<td>24.49</td>
</tr>
<tr>
<td>Baseline</td>
<td>32.48</td>
<td>0.9575</td>
<td>10.15</td>
</tr>
<tr>
<td>+ ADV</td>
<td>30.37</td>
<td>0.9355</td>
<td>15.47</td>
</tr>
<tr>
<td>NAFNet</td>
<td>32.87</td>
<td>0.9606</td>
<td>6.67</td>
</tr>
<tr>
<td>+ ADV</td>
<td>29.91</td>
<td>0.9291</td>
<td>17.33</td>
</tr>
<tr>
<td>Intermediate</td>
<td>29.93</td>
<td>0.9289</td>
<td>6.0224</td>
</tr>
<tr>
<td>+ ADV</td>
<td>29.00</td>
<td>0.9154</td>
<td>24.02</td>
</tr>
<tr>
<td>Intermediate + ReLU</td>
<td>30.39</td>
<td>0.9349</td>
<td>13.87</td>
</tr>
<tr>
<td>+ ADV</td>
<td>28.49</td>
<td>0.9072</td>
<td>23.90</td>
</tr>
</tbody>
</table>
Figure A1. Comparing images reconstructed by all models after CosPGD attack.
Figure A2. Comparing images reconstructed by all models after PGD attack.