Robustness of Deep Neural Networks to Occlusion

Ananya Alekar

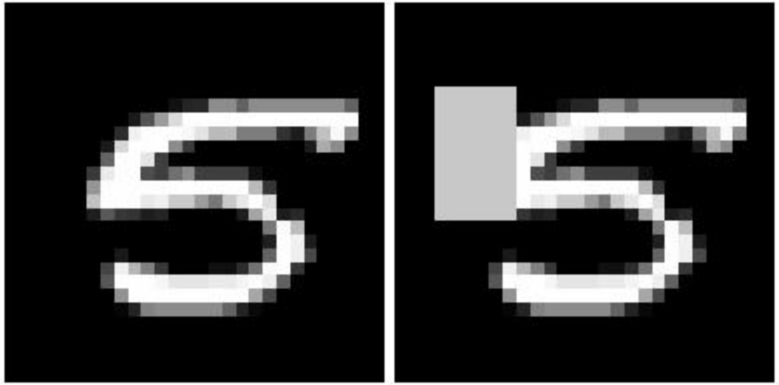
School of Mathematics and Computer Science

Advisor: Prof. Sudakshina Dutta

Occlusion

An occlusion is an event wherein parts of an image are blocked, either partially or completely by another object in the scene.

Types of Occlusion



(a) Uniform

Colour is same in all occluded pixels.

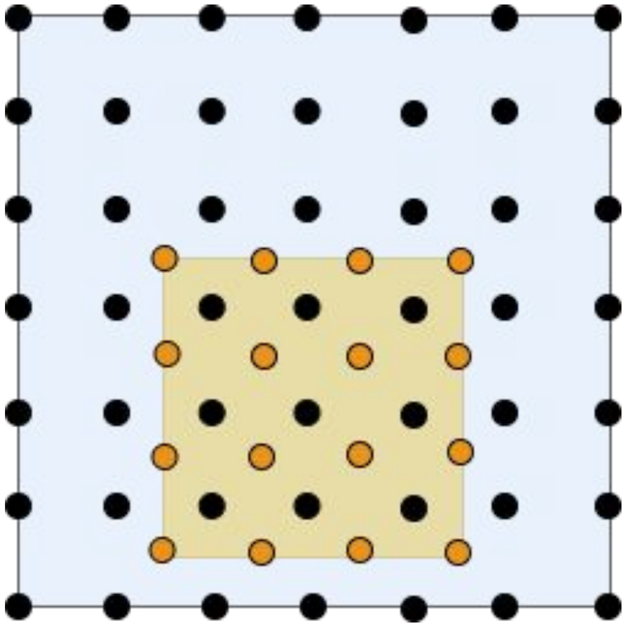
(b) Multiform

Colours vary from $[-\epsilon,\epsilon]$ where ϵ denotes the threshold between original pixel value and occluded pixel value.

Occlusion Position

- L1 distance between image pixel and surrounding occluding pixels is less than 1.
- Contributions from all four surrounding pixels is summed up.

$$s_{ij} = max(0, \sum_{i' \in \mathbb{I}_n} (|i - i' + 1|) + \sum_{j' \in \mathbb{I}_n} (|j - j' + 1|) - 1)$$



Occlusion Operation

$$x'_{ij} = x_{ij} - s_{ij} \times (x_{ij} - \zeta(x, i, j))$$

$$x_{i,j} : \ Original\,pixel\,value$$

$$x'_{i,j} : \ Updated\,pixel\,value$$

$$s_{i,j} : \ Coefficient\,of\,occlusion$$

$$\zeta\,(x, i,\ j) : \ Colour\,of\,occlusion\,at\ \ (i,\ j)$$

Uniform occlusion: $\zeta(x, i, j) \ = \ \mu.$

Multiform occlusion: $\zeta(x, i, j) = x_{ij} + \Delta_p \quad \Delta_p \in [-\epsilon, \epsilon]$

Objective

To explore various occlusion encodings for benchmark datasets like MNIST, CIFAR-10 and GTSRB and verify the robustness of neural networks to these perturbations.

Random Erasing

- Randomly picks upper left corner of occlusion rectangle.
- Replaces or “erases” occlusion rectangle with random colour.

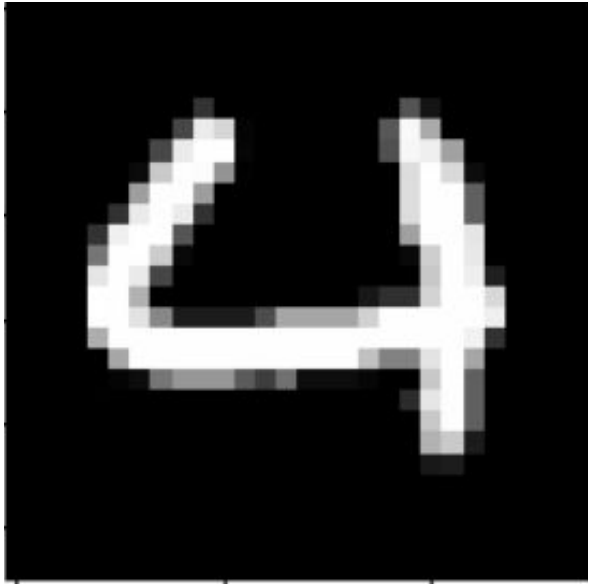
Limitation: Only picks integer coordinates
We need to consider real-valued coordinates.

Algorithm 1: Random Erasing Procedure

Input : Input image I; Image size W and H; Area of image S; Erasing probability p; Erasing area ratio range s_l and s_h; Erasing aspect ratio range r_1 and r_2.
Output: Erased image I^*.
Initialization: $p_1 \leftarrow$ Rand (0, 1).

if $p_1 \geq p$ **then**
  $I^* \leftarrow I$;
  return I^*.
else
  while *True* **do**
    $S_e \leftarrow \text{Rand}\ (s_l, s_h) \times S$;
    $r_e \leftarrow \text{Rand}\ (r_1, r_2)$;
    $H_e \leftarrow \sqrt{S_e \times r_e},\ W_e \leftarrow \sqrt{\frac{S_e}{r_e}}$;
    $x_e \leftarrow \text{Rand}\ (0, W),\ y_e \leftarrow \text{Rand}\ (0, H)$;
    if $x_e + W_e \leq W$ *and* $y_e + H_e \leq H$ **then**
      $I_e \leftarrow (x_e, y_e, x_e + W_e, y_e + H_e)$;
      $I(I_e) \leftarrow \text{Rand}\ (0, 255)$;
      $I^* \leftarrow I$;
      return I^*.
    end
  end
end

Random Erasing

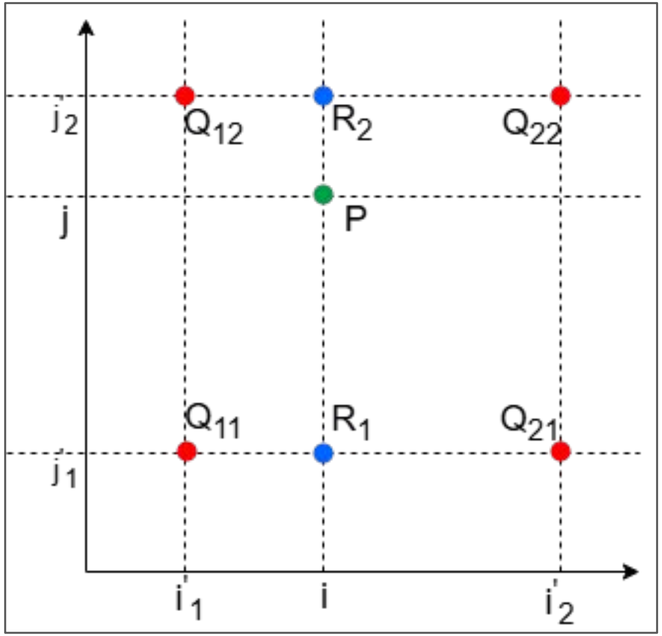


Original Image

Occluded image

Bilinear Interpolation

- Calculates pixel intensities when image is mapped to another geometry.
- Bilinear interpolation considers 4 nearest neighbors of interpolated point.
- Image pixel is affected by occlusion pixel if they are less than √2 units apart.

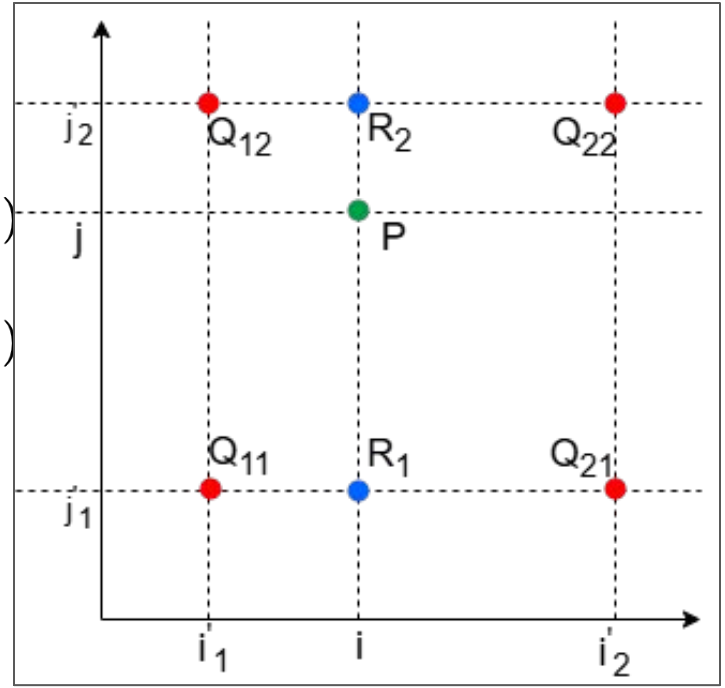


Bilinear Interpolation

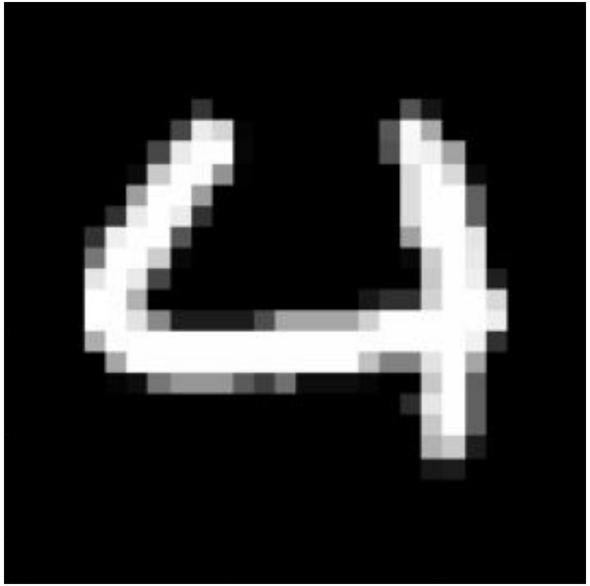
$$R_1(x, y) = Q_{11}(x_2 - x)/(x_2 - x_1) + Q_{21}(x - x_1)/(x_2 - x_1)$$

$$R_2(x, y) = Q_{12}(x_2 - x)/(x_2 - x_1) + Q_{22}(x - x_1)/(x_2 - x_1)$$

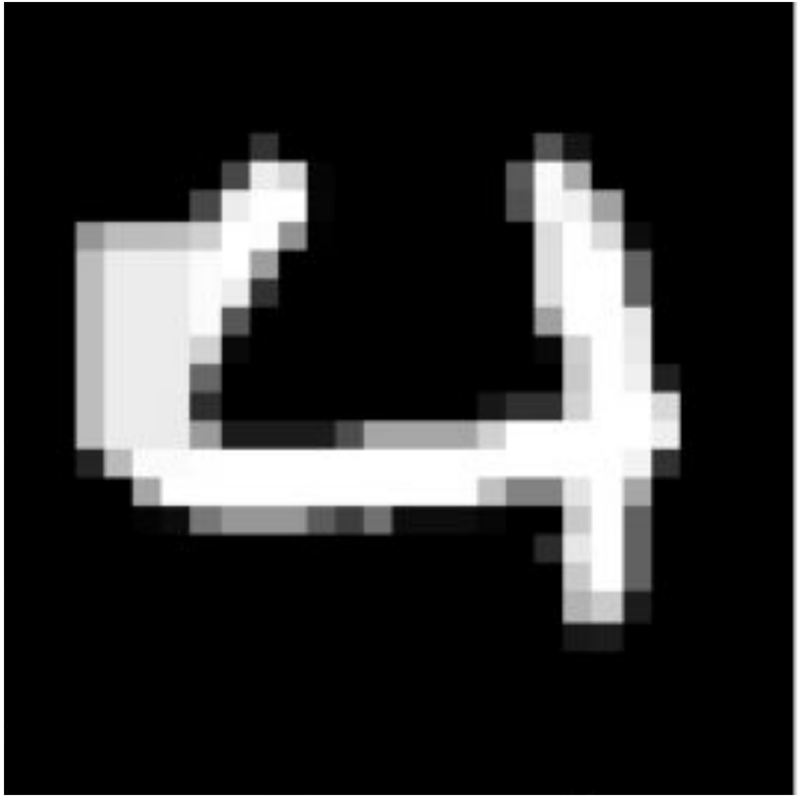
$$P(x, y) = R_1(y_2 - y)/(y_2 - y_1) + R_2(y - y_1)/(y_2 - y_1)$$



Bilinear Interpolation

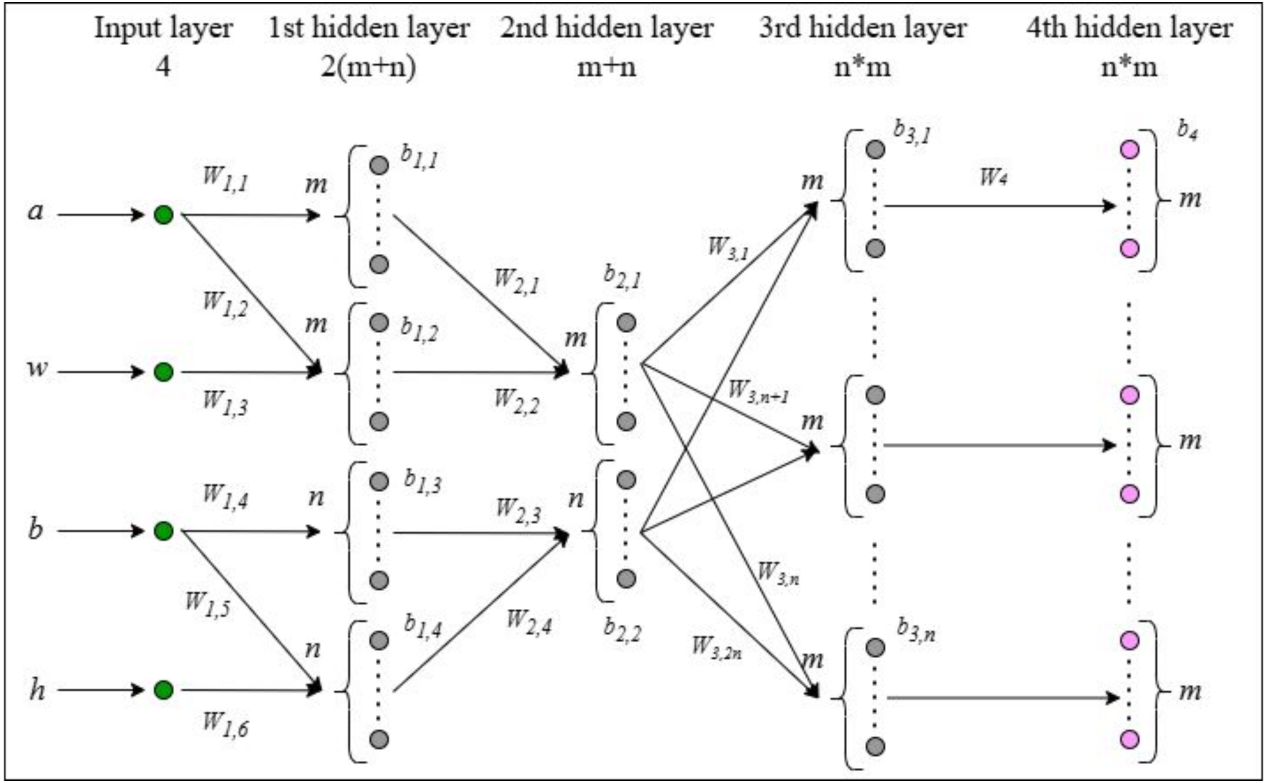


Original Image



Occluded image

Occlusion Encoding with Neural Networks

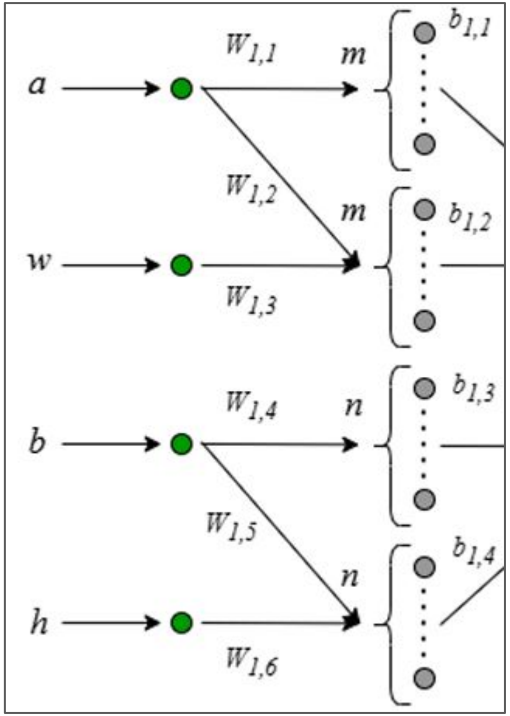


Occlusion Encoding with Neural Networks

First layer encodes the input (a, w, b, h)

$$W_{1,1} = \begin{bmatrix} 1 \\ 1 \\ . \\ . \\ 1 \end{bmatrix}_{m\times 1}, W_{1,2} = \begin{bmatrix} -1 \\ -1 \\ . \\ . \\ -1 \end{bmatrix}_{m\times 1}, W_{1,3} = \begin{bmatrix} -1 \\ -1 \\ . \\ . \\ -1 \end{bmatrix}_{m\times 1};$$

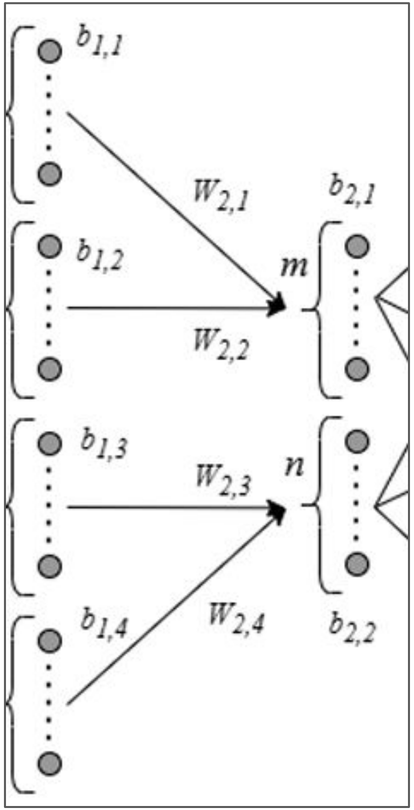
$$b_{1,1} = \begin{bmatrix} -1 \\ -2 \\ . \\ . \\ -m \end{bmatrix}_{m\times 1}, b_{1,2} = \begin{bmatrix} 2 \\ 3 \\ . \\ . \\ m+1 \end{bmatrix}_{m\times 1}$$



Occlusion Encoding with Neural Networks

Second layer: if i^{th} neuron in the first m neurons is 1 and j^{th} neuron in the next n neurons is 1 then (i,j) is occluded.

$$W_{2,i} = \begin{bmatrix} -1 & & & & \\ & -1 & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & -1 \end{bmatrix}_{m \times m}$$

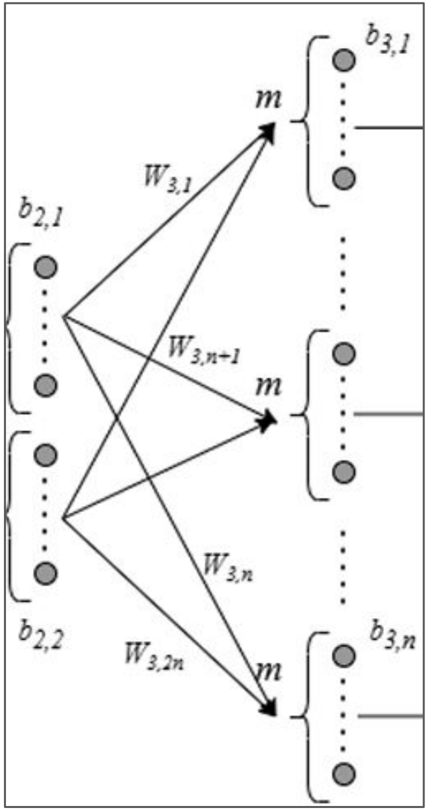


Occlusion Encoding with Neural Networks

Third layer: Outputs m×n neurons, each neuron has the occlusion factor s_{ij} of each pixel.

$$W_{3,i} = \begin{array}{c} \begin{bmatrix} 1 & 0 & . & . & 0 \\ 1 & 0 & . & . & 0 \\ . & & & & . \\ . & & & & . \\ 1 & 0 & . & . & 0 \end{bmatrix}_{m\times m} \\ \quad i \qquad\qquad\qquad \end{array}, W_{3,n+i} = \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & . & & \\ & & & . & \\ & & & & 1 \end{bmatrix}_{m\times n}$$

$$b_{3,i} = \begin{bmatrix} -1 \\ -1 \\ . \\ . \\ -1 \end{bmatrix}_{1\times m}$$



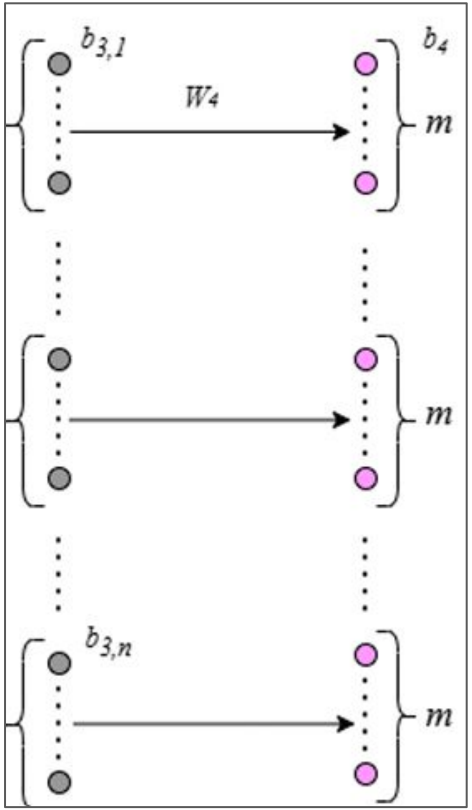
Occlusion Encoding with Neural Networks

Fourth layer: Encodes the pixel values of original image and occlusion colours.

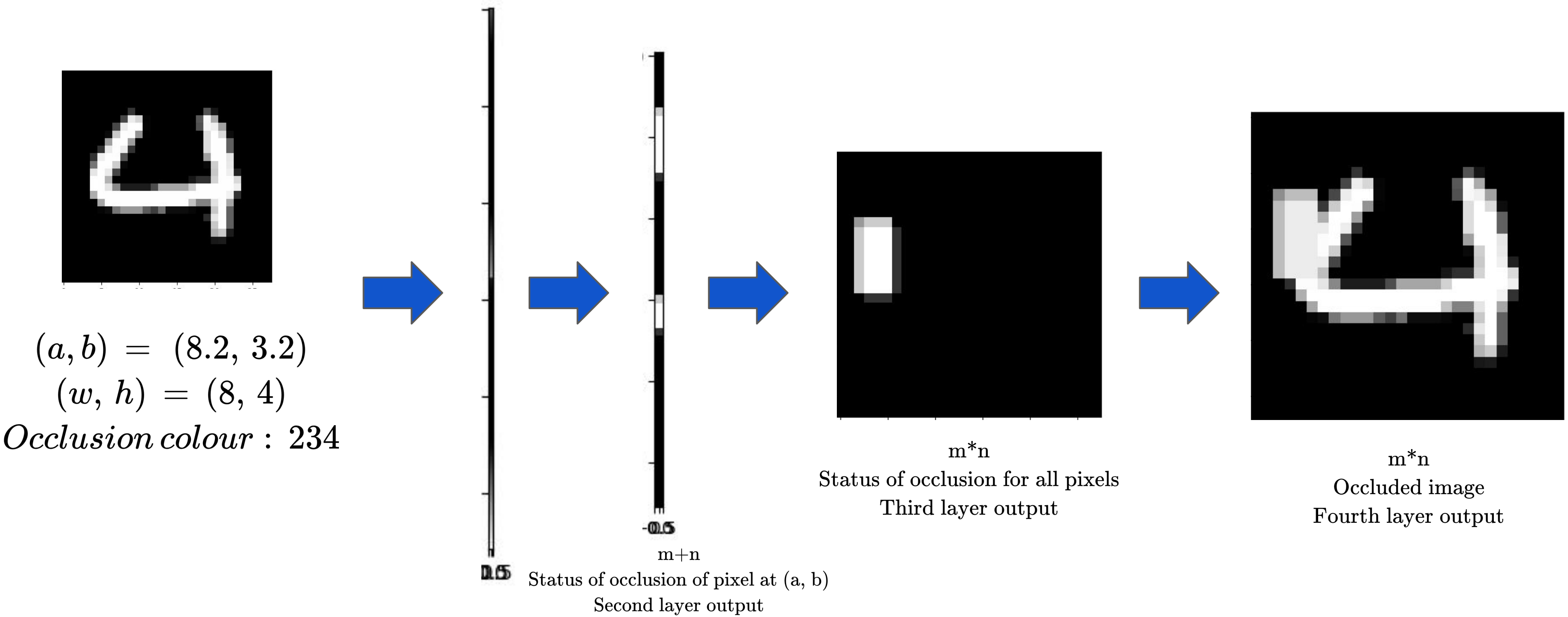
$$W_4 = \begin{bmatrix} \mu - x_1 & & & & \\ & \mu - x_2 & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & \mu - x_{m\times n} \end{bmatrix}_{(mn)\times(mn)} \qquad W_4 = \begin{bmatrix} \Delta_1 & & & & \\ & \Delta_2 & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & \Delta_{mn} \end{bmatrix}_{(mn)\times(mn)}$$

$$b_4 = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ x_{mn} \end{bmatrix}_{mn\times 1}$$

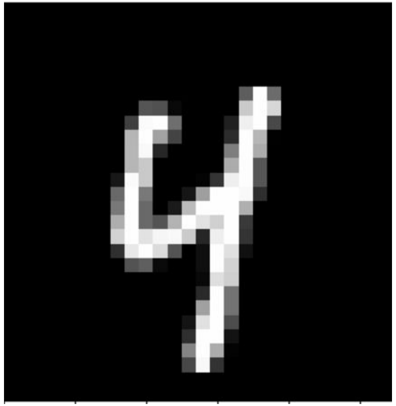
$$W_4 \cdot O_3 + b_4 = (\zeta(x,i,j) - x_{ij}) \times s_{ij} + x_{ij}$$

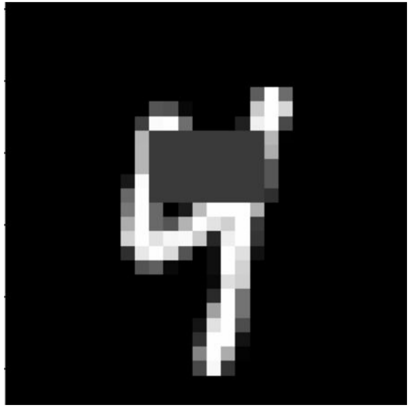


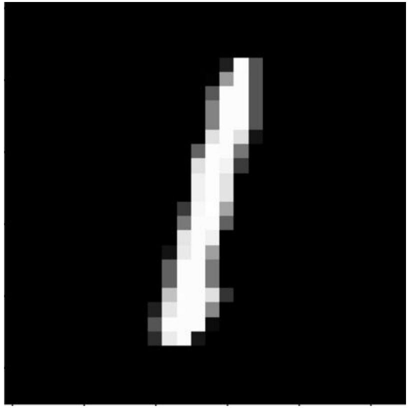
Occlusion Encoding with Neural Networks

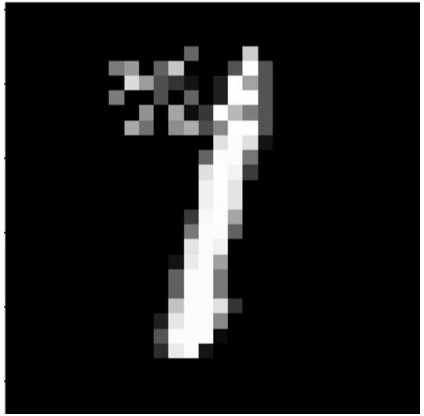


Results









Original image Occluded Image Original image Occluded Image

“4” classified as “9” “1” classified as “8”

Results

Original image

Occluded Image

“deer” classified as “bird”

Original image

Occluded Image

“truck” classified as “ship”

Results




Original image   Occluded Image

“50” classified as “80”

Occluded image   Original Image

“70” classified as “30”

Future Work

- Extend encoding to handle more complex shapes like parallelograms, triangles and circles.
- Estimate a range for the threshold ϵ.

References

1. Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks, 2014.
2. Xiaowei Huang, Marta Kwiatkowska, Sen Wang, and Min Wu. Safety verification of deep neural networks, 2017.
3. [Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks, 2016.
4. [Earl J. Kirkland. Bilinear Interpolation, pages 261–263. Springer US, Boston, MA, 2010.
5. Xingwu Guo, Ziwei Zhou, Yueling Zhang, Guy Katz, and Min Zhang. Occrob: Efficient smt-based occlusion robustness verification of deep neural networks, 2023.
6. Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation, 2017.
7. x engineer. Bilinear interpolation.
8. Yann LeCun and Corinna Cortes. The mnist database of handwritten digits. 2005.
9. Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.
10. Sebastian Houben, Johannes Stallkamp, Jan Salmen, Marc Schlipsing, and Christian Igel. Detection of traffic signs in real-world images: The german traffic sign detection benchmark. In The 2013 International Joint Conference on Neural Networks (IJCNN), pages 1–8, 2013.

Thank You