Exercise 10 - Solution

Network Interpretability and Visualization

Reliable and Interpretable Artificial Intelligence
ETH Zurich

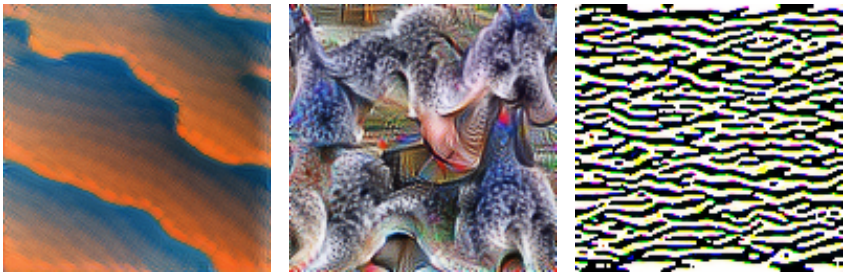

Figure 1: Images strongly activating convolutional filters in three layers created by optimization. Order permuted.

Problem 1 (Feature Visualization)**.** Figure 1 shows three images created by optimization (with Lucid[1]). These images strongly activate convolutional filters in different layers in a convolutional neural network. Order the images according to their depth in the neural network. Provide an argument to justify your answer.

Solution 1. Figure 2 shows the solution. We observe that the first image is a very simple filter for edges in the image. The second image shows some red-blue texture diagonal lines in it. The last image shows an abstract concept like some from of abstract scaled animal. While the interpretations here might be subjective, the overall insight that earlier layers produce primitive filters and later layers high level concepts is clear.

[1]Googles framework for neural network visualization. `https://github.com/tensorflow/lucid`

Figure 2: Images strongly activating convolutional filters in three layers created by optimization.. Order by their position in the network. Each shows the activation of channel zero in the post-ReLU layers `conv2d0`, `mixed3a`, `mixed5a` in GoogLeNet respectively.

Problem 2 (Shapley Values)**.** Consider the classifier $g : \mathbb{R}^3 \to \mathbb{R}^2$ and data points $\boldsymbol{x}$ and $\boldsymbol{x}'$.

$$\boldsymbol{x} := \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \qquad \boldsymbol{x}' := \begin{pmatrix} 1 \\ -2 \\ 3 \end{pmatrix}$$
$$g(\boldsymbol{x}) := \boldsymbol{B}\,\mathrm{ReLU}(\boldsymbol{A}\boldsymbol{x})$$
$$A := \begin{pmatrix} 3 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad B := \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

1. Compute the Shapley Values for all components of both inputs and both target classes. Set non-included features to zero. Discuss the results. (Consider implementing repetitive calculation in python.)

2. Argue why Shapley Values are impractical (applied directly; without approximations) for large images.

Solution 2.

1. Since we have a fixed input and relatively simple neural network we can simplify $g(\boldsymbol{x})$: For the first logit we obtain $g(\boldsymbol{x})_1 = \mathrm{ReLU}(3\boldsymbol{x}_1) + \mathrm{ReLU}(-2\boldsymbol{x}_2)$ and for the second $g(\boldsymbol{x})_2 = \mathrm{ReLU}(-2\boldsymbol{x}_2) + \mathrm{ReLU}(1\boldsymbol{x}_3)$. Plugging in the values for $\boldsymbol{x}$ we see that the first logit is directly proportional to the first component of the input and the second logit to the third component of the input. To calculate the Shapley Value for the first input $\boldsymbol{x}_0$ we let $P := \{\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3\}$ and set $f(S) := g(\boldsymbol{x}_S)_1$ the value of the first logit, where $\boldsymbol{x}_S$ denotes the vector $\boldsymbol{x}$ with components not included in S

set to 0. Thus, for the first component $\boldsymbol{x}_1$ we calculate:

S	$\emptyset$	$\{\boldsymbol{x}_2\}$	$\{\boldsymbol{x}_3\}$	$\{\boldsymbol{x}_2, \boldsymbol{x}_3\}$
$\frac{\lvert S\rvert!(\lvert P\rvert-\lvert S\rvert-1)!}{\lvert P\rvert!}$	$\frac{2}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{2}{6}$
$f(S)$	0	0	0	0
$f(S \cup \{\boldsymbol{x}_1\})$	3	3	3	3

Plugging these values in the formula for Shapley Values, we obtain $C^1_{\boldsymbol{x}_1} = 3$. Similarly we can obtain $C^1_{\boldsymbol{x}_2} = C^1_{\boldsymbol{x}_3} = 0$. We also provide code for the computation `solution10.py` (where indices are zero-based).

For the second logit we obtain:

$$C^2_{\boldsymbol{x}_1} = 0, \quad C^2_{\boldsymbol{x}_2} = 0, \quad C^2_{\boldsymbol{x}_3} = 3.$$

A feature of Shapley values is that the attributions (C) add up to the original function value. For $g(\boldsymbol{x})$ both logits are 3 and they are only influenced by the first and last feature respectively thus we obtain $C^1_{\boldsymbol{x}_1} = C^2_{\boldsymbol{x}_3} = 3$ and zero for all other attributions. Nothing is attributed to $\boldsymbol{x}_2$ as the feature is discarded by the ReLU.

In the case of $\boldsymbol{x}'$, where both logits attain value 7, we get a similar picture, but now the second feature also influences the result:

$$C^1_{\boldsymbol{x}'_1} = 3 \qquad C^2_{\boldsymbol{x}'_1} = 0$$
$$C^1_{\boldsymbol{x}'_2} = 0 \qquad C^2_{\boldsymbol{x}'_2} = 4$$
$$C^1_{\boldsymbol{x}'_3} = 4 \qquad C^2_{\boldsymbol{x}'_3} = 3$$

2. To compute the attribution we need to consider $2^{|P|-1}$ subsets per feature. And while a lot of the computations can be reused for different features this quickly becomes very expensive. Even for a single MNIST image ($28 \times 28 = 784$ pixels) this results in 784×2^{783} subsets to be considered, which without approximations is clearly intractable.

Problem 3 (Robust Features vs Non-Robust Features)**.** (Adapted from [1])

Consider a distribution $\mathcal{D}$ of data points $(\boldsymbol{x}, y) \sim \mathcal{D}$ where

$$y \overset{\text{u.a.r.}}{\sim} \{-1, 1\}, \quad \boldsymbol{x} \in \mathbb{R}^{d+1}, \quad \boldsymbol{x}_1 = \begin{cases} +y & \text{w.p. } p \\ -y & \text{w.p. } 1-p \end{cases}, \quad \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{d+1} \sim \mathcal{N}(\eta y, 1).$$

Here, $\mathcal{N}(\mu, \sigma^2)$ denotes the Gaussian distribution with mean μ and variance σ^2. Further consider the family of linear classifiers $f_i(\boldsymbol{x}) := \text{sign}((\boldsymbol{w}^i)^T \boldsymbol{x})$. We use two different instantiation with $\boldsymbol{w}^1 := (0, \frac{1}{d}, \ldots, \frac{1}{d})^T \in \mathbb{R}^{d+1}$ and $\boldsymbol{w}^2 := (1, 0, \ldots, 0)^T \in \mathbb{R}^{d+1}$. In this example we will use $\eta = \frac{3}{\sqrt{d}}$ and $p = 0.9975$. Thus, we refer to the first component of $\boldsymbol{x}_1$ as a robust feature (as it is strongly associated with y), and $\boldsymbol{x}_2, \ldots, \boldsymbol{x}_{d+1}$ as weak or non-robust features as for large d (e.g., $d \geq 100$) there is only weak correlation y and the individual features.

1. Compute the expected accuracy of f_1, f_2. That is, determine value of $\underset{(\boldsymbol{x},y)\sim\mathcal{D}}{\mathcal{P}}[f(\boldsymbol{x}) = y]$.

2. Consider the data point $y = 1$, $\boldsymbol{x} = (1, 0.1, \ldots 0.1)^T$. Attempt to find an adversarial example $\boldsymbol{x}'$ with $\|\boldsymbol{x} - \boldsymbol{x}'\|_\infty \leq 0.15$, such that it gets misclassified ($f(\boldsymbol{x}') = -1$). For both, f_1 and f_2, find such an example or describe why it does not exist.

Solution 3.

1. We start by considering f_1.

$$\begin{aligned}
&\underset{(\boldsymbol{x},y)\sim\mathcal{D}}{\mathcal{P}}[f_1(\boldsymbol{x}) = y] = \underset{(\boldsymbol{x},y)\sim\mathcal{D}}{\mathcal{P}}\left[\text{sign}((\boldsymbol{w}^1)^T\boldsymbol{x}) = y\right] = \underset{(\boldsymbol{x},y)\sim\mathcal{D}}{\mathcal{P}}\left[y(\boldsymbol{w}^1)^T\boldsymbol{x} > 0\right] \\
&= \underset{(\boldsymbol{x},y)\sim\mathcal{D}}{\mathcal{P}}\left[y\sum_{i=1}^{d+1} \boldsymbol{w}_i^1 \boldsymbol{x}_i > 0\right] = \mathcal{P}_y\left[y\sum_{i=1}^{d+1} \boldsymbol{w}_i^1 \mathcal{N}(\eta y, 1) > 0\right] = \mathcal{P}_y\left[y\sum_{i=2}^{d+1} \frac{1}{d}\mathcal{N}(\eta y, 1) > 0\right] \\
&= \mathcal{P}_y\left[\frac{y}{d}\sum_{i=1}^{d} \mathcal{N}(\eta y, 1) > 0\right] = \mathcal{P}_y\left[\frac{y}{d}\mathcal{N}(d\eta y, d) > 0\right] = \mathcal{P}_y\left[\frac{1}{d}\mathcal{N}(d\eta y^2, dy^2)\right] \\
&\overset{(*)}{=} \mathcal{P}\left[\frac{1}{d}\mathcal{N}(d\eta, d) > 0\right] = \mathcal{P}\left[\mathcal{N}(\frac{d\eta}{d}, \frac{d}{d^2}) > 0\right] = \mathcal{P}\left[\mathcal{N}(\eta, \frac{1}{d}) > 0\right] = 1 - \mathcal{P}\left[\mathcal{N}(-\eta, \frac{1}{d}) \leq 0\right] \\
&= 1 - \Phi\left(\frac{0 - (-\eta)}{\sqrt{\frac{1}{d}}}\right) = 1 - \Phi\left(\sqrt{d}\eta\right) = 1 - \Phi(3) = 0.9987
\end{aligned}$$

   Here Φ denotes the CDF of the standard normal distribution. The equality $(*)$ holds as $y^2 = 1$ for $y \overset{\text{u.a.r.}}{\sim} \{-1, 1\}$, as we use, indicated by $\mathcal{P}_y$. Thus for $\eta = \frac{3}{\sqrt{d}}$ the accuracy $\underset{(\boldsymbol{x},y)\sim\mathcal{D}}{\mathcal{P}}[f_1(\boldsymbol{x}) = y]$ attains 0.9987 independent of d.

For f_2:

$$\mathop{\mathcal{P}}_{(\boldsymbol{x},y)\sim\mathcal{D}}[f_2(\boldsymbol{x}) = y] = \mathop{\mathcal{P}}_{(\boldsymbol{x},y)\sim\mathcal{D}}\left[\text{sign}((\boldsymbol{w}^2)^T\boldsymbol{x}) = y\right] = \mathop{\mathcal{P}}_{(\boldsymbol{x},y)\sim\mathcal{D}}[\text{sign}(\boldsymbol{x}_1) = y] = p = 0.9975$$

Thus the classifier f_2, relying just on the first feature (x_1) attains slightly lower accuracy (0.9975) than the classifier f_1 relying on all the "weak" features ($\boldsymbol{x}_2, \ldots, \boldsymbol{x}_{d+1}$), which attains 0.9987 independently of the value of d. So we can see that depending on η and p the classifier relying on non-robust features $\boldsymbol{x}_2, \ldots, \boldsymbol{x}_{d+1}$ can be more accurate than the classifier relying just on the robust features.

2. Here we will apply FGSM to attempt to compute the adversarial examples. To this end we consider the gradient, given by $\nabla_{\boldsymbol{x}}\boldsymbol{w}^T\boldsymbol{x} = \boldsymbol{w}$.

   Thus for f_1 we find $\boldsymbol{x}' = \boldsymbol{x} - 0.15\,\text{sign}(\boldsymbol{w}^1) = \boldsymbol{x} - (0, 0.15, \ldots, 0.15) = (1, -0.05, \ldots, -0.05)$, which gets classified to class -1.

   However, for f_2 the same process results in $\boldsymbol{x}' = (0.85, 0.1, \ldots, 0.1)$, which still classifies to call 1. In fact, f_2 just considered the first component of $\boldsymbol{x}$, which has value 1. Since we can only change this by 0.15, we can not flip its sign. Thus we can not find an adversarial example for $\epsilon = 0.15$ for f_2.

   So, while f_1 has a higher standard accuracy, f_2 is more robust (and has a higher robust accuracy).

References

[1] Dimitris Tsipras et al. "Robustness May Be at Odds with Accuracy". In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL: https://openreview.net/forum?id=SyxAb30cY7.