Supplementary: Influence Decompositions For Neural Network Attribution

1 Axiomatic Evaluations

In many attribution papers, a measures ability to satisfy certain axioms takes center stage. Here we show which axioms our measure satisfies:

1.1 Sensitivity

The sensitivity axiom introduced in [1] has two forms. The first states that every input/baseline which differ in only one coordinate and have different predictions should yield non-zero attribution in that coordinate. If we assume $k = 1$, and the baseline corresponds to a particular adversarial perturbation at some ϵ that leads to a change in prediction, then our method satisfies this axiom. Additionally, our method can handle cases where multiple coordinates differ for different values of k. The second form of the sensitivity axiom states that neural network functions which don't mathematically depend on a variable should assign 0 attribution to this variable. Again, if $k = 1$ our approach satisfies this axiom, since no adversarial perturbation will affect the outcome.

1.2 Implementation Invariance

Implementation invariance [1] states that two neural networks which are functionally equivalent should have the same attributions. This is true for our approach, since applying the same input perturbations to both networks will always yield the same output if the functions are truly functionally equivalent.

1.3 Input Invariance

Input invariance [2] states that for networks $f_1 = f(x)$ and $f_2 = g(x + c)$, if $f_1 = f_2$, then their attributions should be the same. This axiom is satisfied by our method, since applying the same adversarial perturbation in both cases will produce the same output. This result is consistent with similar measures to ours [3].

1.4 Symmetry Preserving

Symmetry preserving [1] states that any two input variables that can be exchanged without affecting the outcome should have the same attribution. As with the axioms above, if we consider $k = 1$, this is true for our approach, and larger values of k generalize the axiom to larger subsets of variables.

1.5 Linearity

Linearity [1] states that attributions for a function f_3 which is a linear combination of other functions (say, f_1 and f_2) should be a linear combination of the attributions for these functions. Since we use misclassification as our main output metric, this axiom is not satisfied. For example f_1 and f_3 may be misclassified while f_2 wasn't, but this doesn't report the effects of the weights. If value of the loss was used in place of misclassification, this axiom would be satisfied.

1.6 Completeness

Completeness [1] states that the total attribution should equal the difference between an image and its baseline. Our approach does not satisfy this axiom, but as we argue in the paper, we do not believe that linear distance to a fixed baseline is a good metric or axiom to go by.

2 Additional Attribution Plots

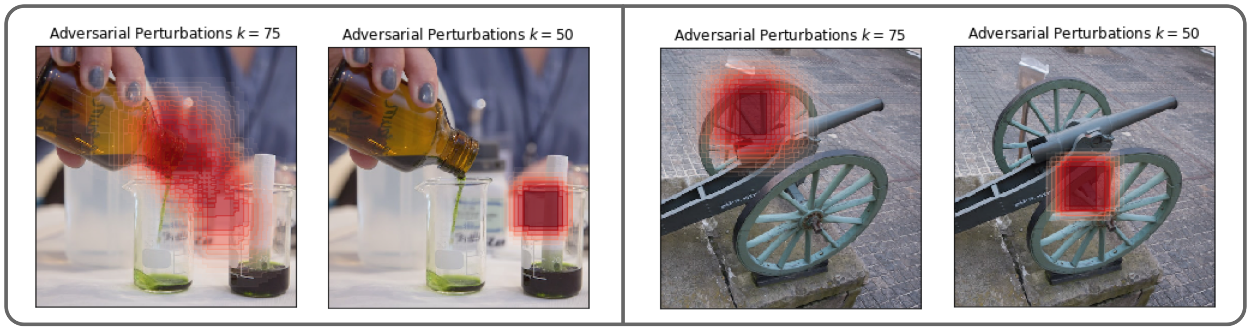
Adversarial Perturbations $k = 75$
Adversarial Perturbations $k = 50$
Adversarial Perturbations $k = 75$
Adversarial Perturbations $k = 50$

Figure 1: Our approach for different values of $k \times k$ contiguous blocks. Value of ϵ was 10.0, and the attack method used was $\mathcal{L}^2$ Fast-Gradient.

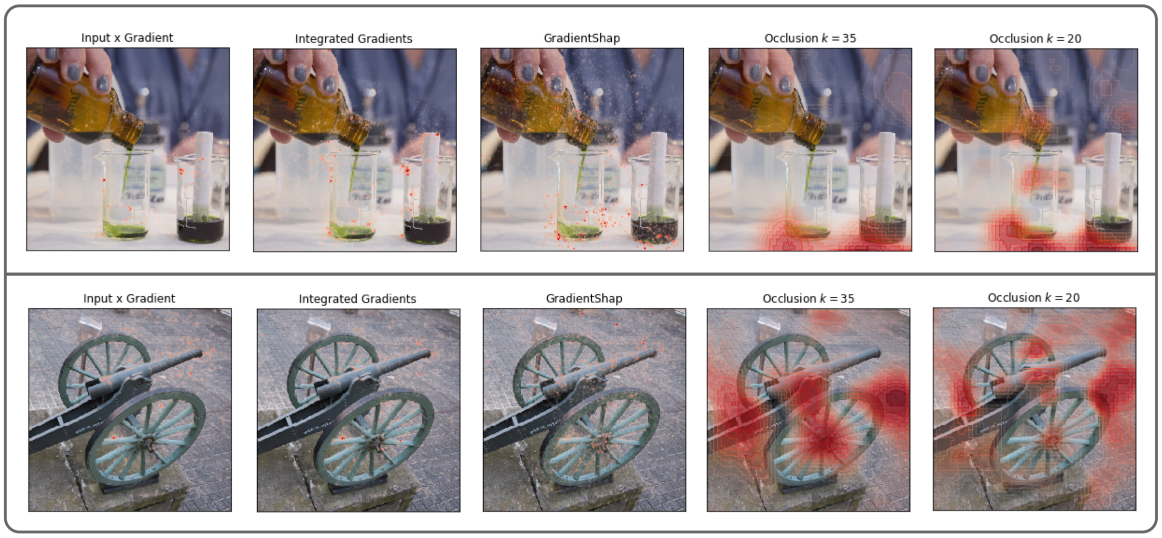

Figure 2: Alternative attribution approaches on the same images.

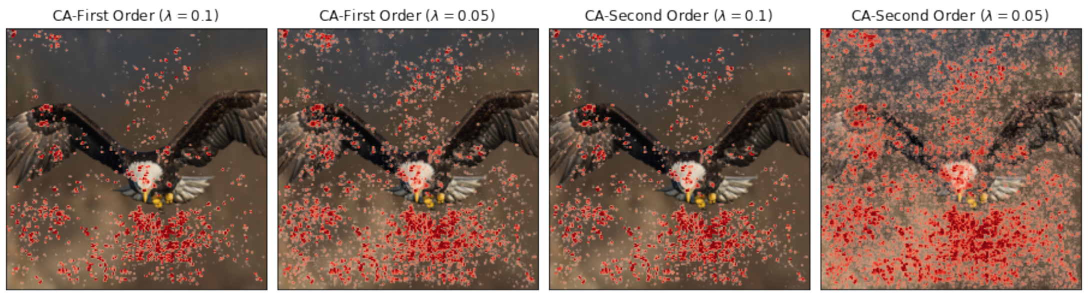
CA-First Order ($\lambda = 0.1$)
CA-First Order ($\lambda = 0.05$)
CA-Second Order ($\lambda = 0.1$)
CA-Second Order ($\lambda = 0.05$)

Figure 3: Attribution plots for first and second order ”content-aware” methods from [4] for various hyperparmeter settings. These approaches attempt to find a single reduced subset of features. This is to contrast our approach, which attempts to quantify the average contribution form a set k features.

3 Comparing Hyperparameters For Adversarial Attacks

Methods	150×150	125×125	100×100	75×75	50×50	25×25
L2PGD ($\epsilon = 0.5$)	1.0	0.632	0.229	0.047	0.0	0.0
L2PGD ($\epsilon = 1.0$)	1.0	0.947	0.608	0.292	0.135	0.0
L2PGD ($\epsilon = 10.0$)	1.0	1.0	0.913	0.514	0.204	0.051
LinfPGD ($\epsilon = 0.1$)	1.0	1.0	0.911	0.047	0.0	0.0
LinfPGD ($\epsilon = 0.5$)	1.0	0.632	0.229	0.047	0.0	0.0
LinfPGD ($\epsilon = 1.0$)	1.0	0.947	0.608	0.292	0.135	0.0
LinfPGD ($\epsilon = 10.0$)	1.0	1.0	0.913	0.514	0.204	0.051
L2BI ($\epsilon = 0.5$)	1.0	0.864	0.435	0.203	0.015	0.0
L2BI ($\epsilon = 1.0$)	1.0	0.977	0.622	0.35	0.17	0.0
L2BI ($\epsilon = 10.0$)	1.0	1.0	0.947	0.585	0.292	0.076
LinfBI ($\epsilon = 0.1$)	1.0	1.0	0.966	0.654	0.195	0.066
LinfBI ($\epsilon = 0.5$)	1.0	1.0	0.954	0.585	0.275	0.112
LinfBI ($\epsilon = 1.0$)	1.0	1.0	0.908	0.62	0.313	0.156
LinfBI ($\epsilon = 10.0$)	1.0	1.0	0.97	0.708	0.366	0.190
L2FG ($\epsilon = 0.5$)	1.0	.852	0.42	0.204	0.032	0.0
L2FG ($\epsilon = 1.0$)	1.0	0.99	0.658	0.357	0.186	0.0
L2FG ($\epsilon = 10.0$)	1.0	1.0	0.942	0.582	0.265	0.09
LinfFG ($\epsilon = 0.1$)	1.0	1.0	0.822	0.477	0.282	0.109
LinfFG ($\epsilon = 0.5$)	1.0	1.0	0.957	0.675	0.357	0.191
LinfFG ($\epsilon = 1.0$)	1.0	1.0	0.979	0.718	0.386	0.197
LinfFG ($\epsilon = 10.0$)	1.0	1.0	0.979	0.718	0.386	0.197

4 Example Neural Network Architectures Implementing Relevant Boolean Functions

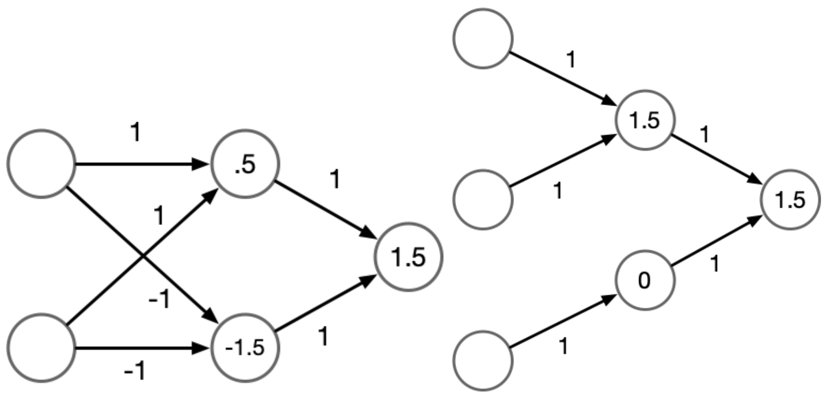

Figure 4: Neural network implementations of XOR (left) and AND (right) gates. These functions can be scaled to higher number of input variables via recursive application. For example, $XOR(1, 2, 3) = XOR(3, XOR(1, 2))$.

References

[1] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," 2017.

[2] P.-J. Kindermans, S. Hooker, J. Adebayo, M. Alber, K. T. Schütt, S. Dähne, D. Erhan, and B. Kim, "The (un)reliability of saliency methods," 2017.

[3] A. Chattopadhyay, P. Manupriya, A. Sarkar, and V. N. Balasubramanian, "Neural network attributions: A causal perspective," 2019.

[4] S. Singla, E. Wallace, S. Feng, and S. Feizi, "Understanding impacts of high-order loss approximations and features in deep learning interpretation," 2019.