6. Appendix

6.1. Null Feature Experiment

More visual examples for the null feature experiment are provided in Fig. 6.

6.2. Feature Saturation Experiment

The visual example for feature saturation is provided in Fig. 5.

6.3. Optimization (Focal Loss) Details

The optimization problems in the framework have multiple losses. Therefore it is a challenge to balance the optimization between the losses. We use the focal loss [10] method to balance the optimization. The following terms each indicate the keys used during optimization phase as focal loss entries.

$$\begin{aligned} \kappa_{\mathcal{L}} &= \sigma_t(\Phi_t(X)) \\ \kappa_{\{f_a\}} &= tanh(\frac{|\Phi_t(X_{\{f_a\}}) - \Phi_t(X_{\{\}})|}{\min(|\Phi_t(X_{\{f_a\}})|, |\Phi_t(X_{\{\}})|)}) \\ \kappa_{\{f_a,\, f_b\}} &= tanh(\frac{|\Phi_t(X_{\{f_a,\, f_b\}}) - \Phi_t(X_{\{f_a\}})|}{\min(|\Phi_t(X_{\{f_a,\, f_b\}})|, |\Phi_t(X_{\{f_a\}})|)}) \end{aligned} \tag{12}$$

where $\sigma_t(.)$ designates softmax function corresponding to the target t. The weighted average of the term $\kappa_{\mathcal{L}}$ is emploied in each of the loss terms as the multiplicand of the first term. The weighted average of other two are used in their corresponding scenarios as the multipicand of the latter terms. The weighted average through each iteration is calculated in the following manner:

$$\hat{\kappa}_{t+1} = \alpha\kappa_t + (1-\alpha)\hat{\kappa}_t \tag{13}$$

During all the experiments, the α is set to 0.1, and the initial value for κ is 0.5. To further faciliate the optimization phase, we initially optmize the features to maximize $\Phi_t(X)$ fo their designated target class.

6.3.1 Comparison to Existing Evaluations

Each evaluation in Sec. 2.2 evaluates explanations from a different perspective. [16, 34, 35] discuss the axioms *theoretically*. Proofs are broken in practice. E.g. our framework identifies issues with DeepSHAP (as also shown in [34]). Our framework is the practical incarnation for the axioms in [16, 34, 35]. [31] provides a class-sensitivity evaluation. Our results complement them. The metric in [31] considers the correlation between maps of different classes, thus identifies gradient as class sensitive. But the low correlation is due to a mere shift in noise, which our method avoids. The pointing game [38] assumes the model uses features that we humans use. We remedy this by having control over generated features (Sec 2.2). GBP, IG get high scores in [25], but we reveal they attribute to null-feature and are class insensitive. [11] aims to evaluate another aspect, feature importance (Sec 2.2 for limitations). A method such as FullGrad and GradCAM++ can highlight important features, but we show they attribute to Null and are class-insensitive. In cases where there is only one highly activating region in the input, these methods will reveal them ([14]).

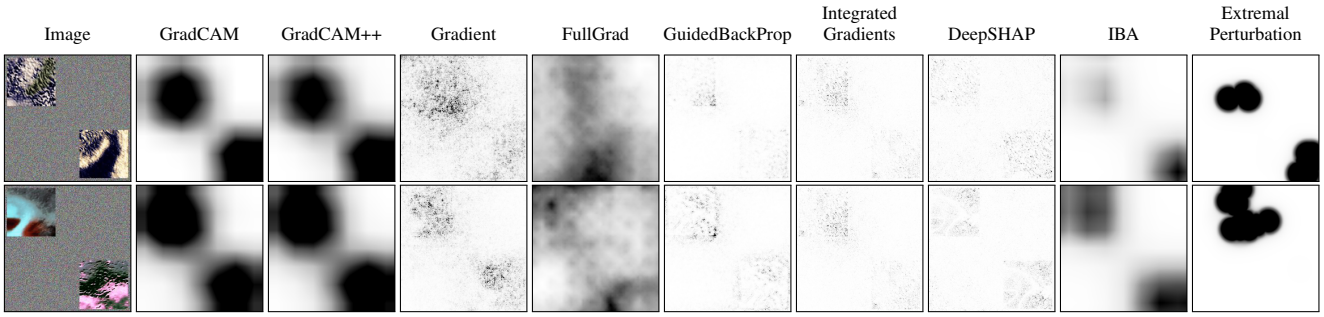

Figure 5. **Feature Saturation Experiment**: Each row is a sample from the feature saturation experiment. In this experiment, the features (patches) each saturate the output. In other terms, each individually generates the same output as their combination. A desired property for the attribution method is to distribute the contribution equally between the features. We observe that Extremal Perturbation and IBA can lean toward attributing the output to only one of the features. The formulation of these two method is based on keeping a region that keeps the output prediction. Thus, it is expected that they lean toward one feature.

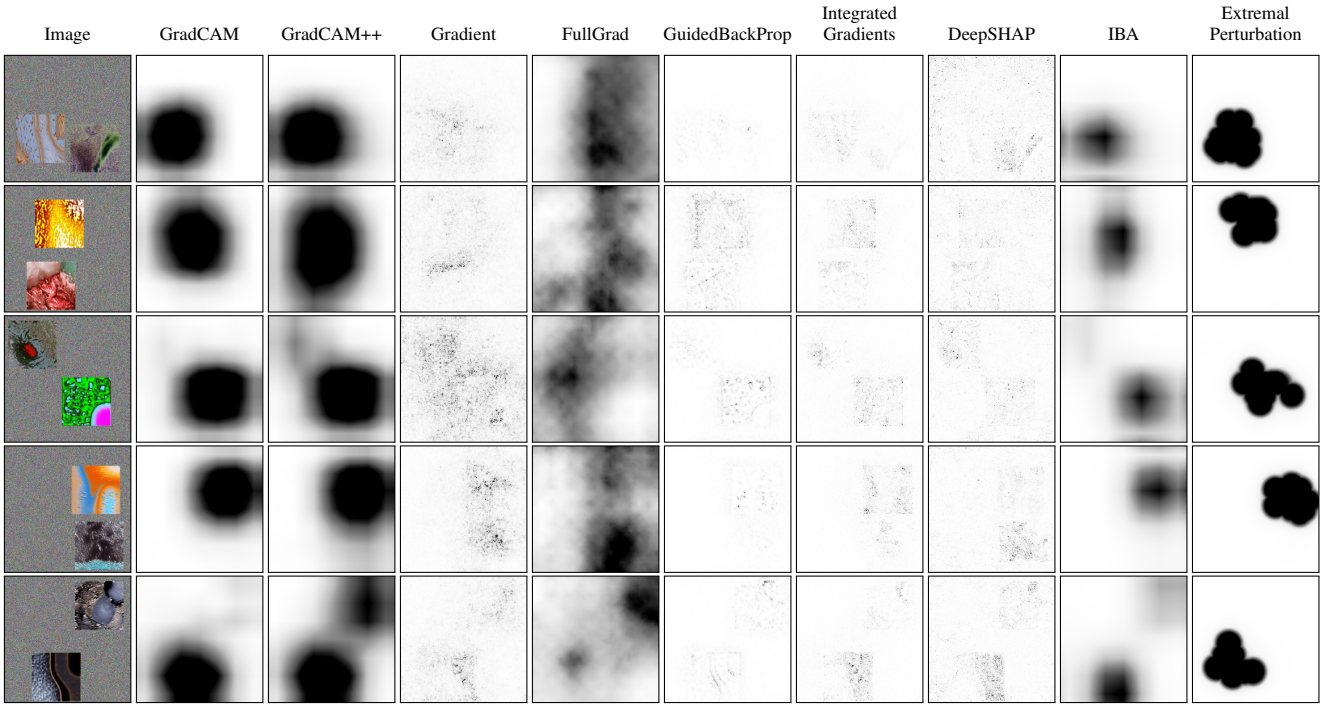

Figure 6. **Null Feature Experiment**: Each row represents a sample from the null feature experiment. In each row, the image on the left represents the generated features on the reference (noise) input. The features are generated using the model itself. Within the image, the lower feature (patch) is generated such that it is a null feature for the output. The rest of the images represent different explanations. As the second feature is a null feature, an explanation method should not assign importance to it. We observe that GradCAM, IBA, and Extremal Perturbation perform best in avoiding the null feature.