Evaluating Feature Importance Estimates

Sara Hooker [1 2] **Dumitru Erhan** [1] **Pieter-Jan Kindermans** [1 2] **Been Kim** [1]

Abstract

Estimating the influence of a given feature to a model prediction is challenging. We introduce **ROAR**, **R**em**O**ve **A**nd **R**etrain, a benchmark to evaluate the accuracy of interpretability methods that estimate input feature importance in deep neural networks. We remove a fraction of input features deemed to be most important according to each estimator and measure the change to the model accuracy upon retraining. The most accurate estimator will identify inputs as important whose removal causes the most damage to model performance relative to all other estimators. This evaluation produces thought-provoking results – we find that several estimators are less accurate than a *random* assignment of feature importance. However, averaging a set of squared noisy estimators (a variant of a technique proposed by Smilkov et al. (2017)), leads to significant gains in accuracy for each method considered and far outperforms such a random guess.

1. Introduction

In a machine learning setting, a question of great interest is estimating the influence of a given input feature on the prediction made by a model. Understanding what input features are important helps improve our models, builds trust in the model prediction and isolates undesirable behavior. Furthermore, a correct estimate is not only desirable but is often required before humans choose to use models in sensitive domains like healthcare, autonomous vehicles and credit scoring. In domains of this nature, determining the accuracy of the estimate is paramount as an incorrect explanation of a model prediction may have intolerable costs on human welfare.

Estimating input importance given a deep neural network (DNN) is particularly challenging because there is typically a high number of input features and we are unable to concisely determine the representation learnt by the model. For example, the model input for a computer vision task is often a digital image and an input feature can be defined as a pixel or a group of connected pixels. Thus, a single image in the dataset can easily be associated with a quarter of a million input features. The aim of feature importance estimators is to quantify the importance of each of these input features to the model prediction for that image. The set of estimates for all pixels in the image is often treated as a sorted ranking of importance, or they are visualized as a natural image "heatmap" to assist humans in understanding parts of the image that the model pays attention to.

Despite the challenges involved, a substantial body of research on input importance estimation and numerous estimators have been proposed (Baehrens et al., 2010; Bach et al., 2015; Zintgraf et al., 2017; Selvaraju et al., 2017; Sundararajan et al., 2017; Simonyan & Zisserman, 2015; Zeiler & Fergus, 2014; Springenberg et al., 2015; Kindermans et al., 2017; Montavon et al., 2017; Fong & Vedaldi, 2017; Dabkowski & Gal, 2017; Zhang et al., 2016a; Shrikumar et al., 2017; Zhou et al., 2014; Ross & Doshi-Velez, 2017). Assessing the correctness of importance rankings is complicated by the lack of ground truth. If we knew the true importance of each pixel in the image to the model prediction, estimating feature importance would reduce to a traditional supervised learning problem. In this case, model performance could be evaluated easily.

Without a clear benchmark to track task performance, comparing the relative merit of an estimator is often based upon human studies (Selvaraju et al., 2017; Ross & Doshi-Velez, 2017) which ask "is this estimate meaningful to a human visual system classifying the same image?" However, asking whether the ranking is meaningful to a human may be entirely different from determining whether this estimate is an accurate reflection of the representation learnt by the model. This work primarily concerns itself with proposing a quantitative benchmark for determining the accuracy of each estimator.

We introduce **ROAR**, **R**em**O**ve **A**nd **R**etrain, a benchmark measuring the *relative* accuracy of input feature importance estimators. For each estimator, we replace a fraction of all

[1]Google Brain [2]Work done as part of the Google Brain AI Residency program.. Correspondence to: Sara Hooker <shooker@google.com>.

2018 ICML Workshop on Human Interpretability in Machine Learning (WHI 2018), Stockholm, Sweden.

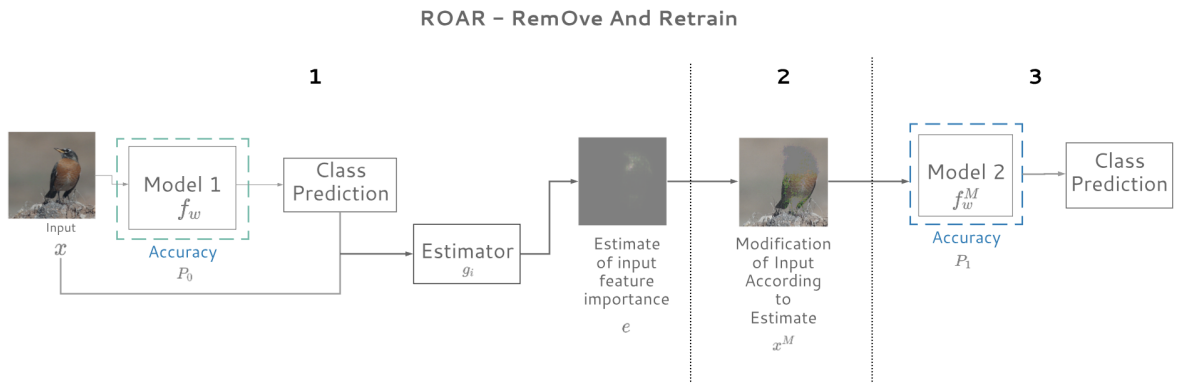

Figure 1. ROAR evaluates the relative accuracy of feature importance estimators. **1)** Firstly, we generate a ranking according to each estimator for each image in the dataset. The ranking is an estimate of each input feature's contribution to a specified output activation given a trained model. **2)** We use this ranking to modify the dataset by removing a fraction of input features estimated to be most important. **3)** Subsequent to this modification of the entire dataset, we train a new model from random initialization on the modified inputs and measure the degradation to model performance. The most accurate estimator will identify input features whose modification causes the most degradation to model performance upon retraining.

ImageNet pixels (Deng et al., 2009) that are estimated to be most important with a constant value that is irrelevant for the classification task (we use the per-channel mean computed using a sample of 2000 images). Fig. 2 shows the modification of a single image example while varying the fraction of all input features that are replaced. We retrain a Resnet-50 model (He et al., 2015) independently on each these modified datasets and we measure the subsequent change to model performance. In Fig. 1, we illustrate the key steps in the ROAR framework.

The most accurate estimator assigns the highest importance to the inputs whose removal causes the largest damage to test-set accuracy relative to all other estimators considered. In addition to comparing the relative accuracy of a set of estimators, we also compare estimator performance to a random assignment of importance. This *random* baseline allows us to answer the question: **is the estimate of importance more accurate than a random guess?**

Training the model from random initialization is *crucial* in order for the constant value for which we replaced the input to be considered "uninformative." Without retraining, decoupling whether the model's degradation in performance is due to the replacement value being outside of the training data manifold or due to the accuracy of the estimate. Model vulnerability to the introduction of "new evidence" has already been widely acknowledged (Dabkowski & Gal, 2017; Fong & Vedaldi, 2017).

Our contributions can be enumerated as follows:

- We introduce ROAR, a benchmark to evaluate the accuracy of a set of estimators. Unlike prior quantitative benchmarks, ROAR evaluates estimator accuracy using a large scale dataset (all of ImageNet). In addition, ROAR, unlike prior perturbation-based benchmarks retrains the model to control for degradation due to the introduction of data points outside of the training manifold.
- We find that the subset of estimators that we evaluate using ROAR are less accurate than a random assignment of feature importance. However, we find that certain transformations of the raw estimate that benefit the accuracy of all estimators considered. For example, averaging and squaring a set of noisy estimates produced by any method (a variant of a technique proposed by Smilkov et al. (2017)) drastically improves the accuracy of the estimate, and the technique far outperforms such a random guess (see the left inset chart of Fig. 5).
- Finally, we show training performance proves surprisingly robust to modifying the majority of all input features. After randomly replacing 90% of all input features, we can still train a model that achieves 63.71 ± 0.13 (average across 5 independent runs). These results suggest that many redundancies in the input feature space exist, however, the basic estimators that we consider are no better than a random guess at identifying them.

2. Related Work

Interpretability research is diverse, and many different approaches are used to gain intuition about the function implemented by a neural network. For example, one can distill or constrain a model into a functional form that is considered

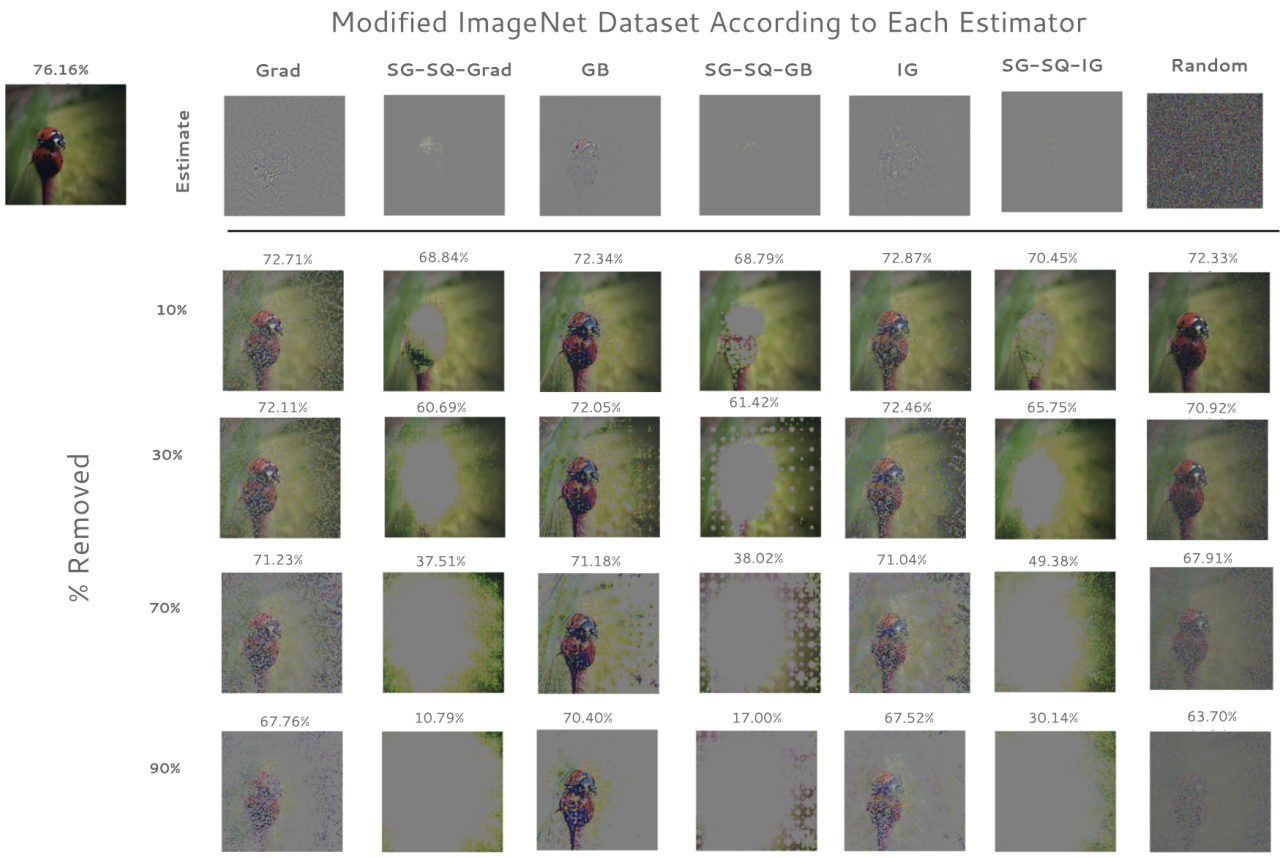

Figure 2. A single example from each dataset modified according to the ROAR framework. The value used to replace inputs that are 'removed' is a constant mean value of a dataset sample of 2000 modified image. Under ROAR, the most accurate estimator will degrade model performance the most for a given fraction of inputs modified. Above each example, we display the average validation set accuracy across three independent training runs. **From left to right**: gradient heatmap (GRAD), SmoothGrad-Squared Grad (SG-SQ-GRAD), Guided BackProp (GB), SmoothGrad-Squared Guided BackProp (SG-SQ-GB), Integrated Gradients (IG) and SmoothGrad-Squared Integrated Gradients (SG-SQ-IG). Transforming an estimator by averaging a set of noisy, squared estimates SG-SQ substantially improves the accuracy of all estimators.

more interpretable (Ba & Caruana, 2014; Frosst & Hinton, 2017; Wu et al., 2017; Ross et al., 2017). Other methods explore the role of neurons or activations in hidden layers of the network (Olah et al., 2017; Raghu et al., 2017; Morcos et al., 2018; Koh & Liang, 2017). Human interface and visualization tools which require a human to determine the starting point and direction of investigation (Kim et al., 2017; van der Maaten & Hinton, 2008) and finally there are also the input feature importance estimators that we evaluate in this work, which estimate the importance of an input feature to a specified output activation.

Various quantitative evaluation frameworks to evaluate input feature importance estimators in deep neural networks have already been proposed. Kindermans et al. (2017) define a unit test that constructs a narrow ground-truth in which invariance to factors that do not affect the model can be measured. Adebayo et al. (2018) randomize layers of the model and measures the change to the estimate. The "pointing game" proposed by Zhang et al. (2016b) and subsequent variations (Selvaraju et al., 2017; Fong & Vedaldi, 2017) evaluate whether the input estimated to be most important lies in a bounded box for the target class in question.

Most relevant to our work are modification based evaluation measures proposed originally by Samek et al. (2017) with subsequent variations (Ancona et al., 2017; Fong & Vedaldi, 2017; Kindermans et al., 2017). In this line of work, one replaces the inputs estimated to be most important with a value considered "meaningless" to the task. These methods measure the subsequent degradation to the *trained* model at inference time.

To the best of our knowledge, unlike prior modification based evaluation measures, our benchmark requires retraining the model from *random initialization* on the modified

dataset rather than re-scoring the modified image at inference time. We do not modify a connected region, or "patch", of connected pixels according to the aggregated estimates of importance. Instead, we simply modify the fraction of inputs estimated to be most important. Finally, we moreover modify every image in ImageNet (1.28 million training and $50,000$ validation images). Most previous evaluations (with the exception of one) have involved modifying less than $6,000$ images). The largest evaluation to date used the ImageNet validation set (Kindermans et al., 2017).

3. Estimating Input Feature Importance

A CNN is trained to approximate the function F that maps an input variable X to an output variable Y, formally $F : X \mapsto Y$. Without loss of generality we represent the image input as a feature vector $\mathbf{x} \in R^N$. $y \in Y$ is a discrete label vector associated with each input x. A given input image $\mathbf{x}$ can be decomposed into a set of pixels $\{x_i\}_{i=1}^N$.

An estimator $\mathcal{G}$ produces a vector of estimates $\mathbf{e}$. $e_i \in \mathbf{e}$ is the estimated importance of x_i to an output activation A_n^l, where l and n designate the layer of the model and the neuron of interest respectively. A_n^l is typically specified to be the maximum pre-softmax score or the softmax probability.

3.1. Evaluation Methodology

We can rank $\mathbf{e}$ into an ordered list $\{e_i^o\}_{i=1}^N$ so that e_0^o corresponds to the input feature x_i estimated to be most important. For a fraction t of this ordered set, we replace the corresponding values in the raw image vector $\mathbf{x}$ with a constant uninformative variable c. We adjust X repeatedly to create a family of distributions $p(y, x^M; t_i, g_i)$, where each distribution is defined by incrementally increasing the fraction of inputs modified $t = [0. : 0.1 : 1]$ and varying the estimator $g_i \subset \mathcal{G}$.

When $t = 1$, $x^M = x$ and the test-set accuracy $\xi(\mathbf{x}^M|g_i)$, will only differ from $\xi(\mathbf{x})$ of a model trained on unmodified inputs by an epsilon term ε that is caused by the natural variation in training performance.

When $t = 0$, we have replaced all input features with the constant value c and learning a representation should not be possible.

In between $t = [0 : .1 : 1]$ we are unable to precisely determine how removing inputs will change the test-set accuracy ξ since we do not know the true distribution of importance a priori. However, we can compare the degradation of test accuracy between estimators for the same fraction t.

The most accurate estimator g_i^* assigns the highest importance to the features whose removal causes the sharpest degradation to model performance ξ. Thus, under ROAR, the most desirable estimator is the one that results in the minimum test accuracy $\xi(\mathbf{x}^M|g^*)$, where $\mathbf{x}^M$ is the modified dataset given the estimator g:

$$\xi(\mathbf{x}^M|g^*) = \min_{g \subset \mathcal{G}} \xi(\mathbf{x}^M|g)$$

In addition, we determine an estimate g to be *better than a random assignment of importance* g^R if the test accuracy trained on the randomly modified inputs is such that:

$$\xi(x^M|g) < \xi(x^M|g^R)$$

3.2. Estimators Considered

In this work, our initial evaluation is constrained to a subset of all possible estimators $\{g_i\} \subset \mathcal{G}$ which we introduce briefly below. We selected this subset based upon the availability of open source code and the ease of implementation on a ResNet-50 architecture. For example, we did not include PatternNet (Kindermans et al., 2017) because an open source implementation does not exist for ResNet-50.

Gradients or Sensitivity heatmaps (Simonyan & Zisserman, 2015; Baehrens et al., 2010) (GRAD) are the gradient of the output activation of interest A_n^l with respect to x_i:

$$\mathbf{e} = \frac{\partial A_n^l}{\partial x}$$

Guided Backprop (Springenberg et al., 2015) (GB) is an example of a signal method. Signal estimators aim to visualize the input patterns that cause the neuron activation A_n^l in higher layers. GB computes this by using a modified backpropagation step in which the flow is stopped when the gradient is less than zero at a ReLu gate. Other examples of signal methods, which we did not evaluate here, include DeConvNet (Zeiler & Fergus, 2014) and PatternNet (Kindermans et al., 2017). The performance of Guided Backprop should not be considered representative of other signal methods, and we welcome the opportunity to benchmark additional methods in future iterations of this work.

Integrated Gradients (Sundararajan et al., 2017) (IG) is an example of an attribution method. Attribution estimators assign importance to input features by decomposing the output activation A_m^l into contributions from the individual input features (Bach et al., 2015; Sundararajan et al., 2017). Attribution methods require that all contributions sum to the activation of interest. This property is often termed "completeness":

$$\sum_{i=1}^{N} e_i = A_n^l.$$

. Integrated gradients interpolate a set of estimates for values of $\alpha \subset [0, 1]$ between a non-informative reference point

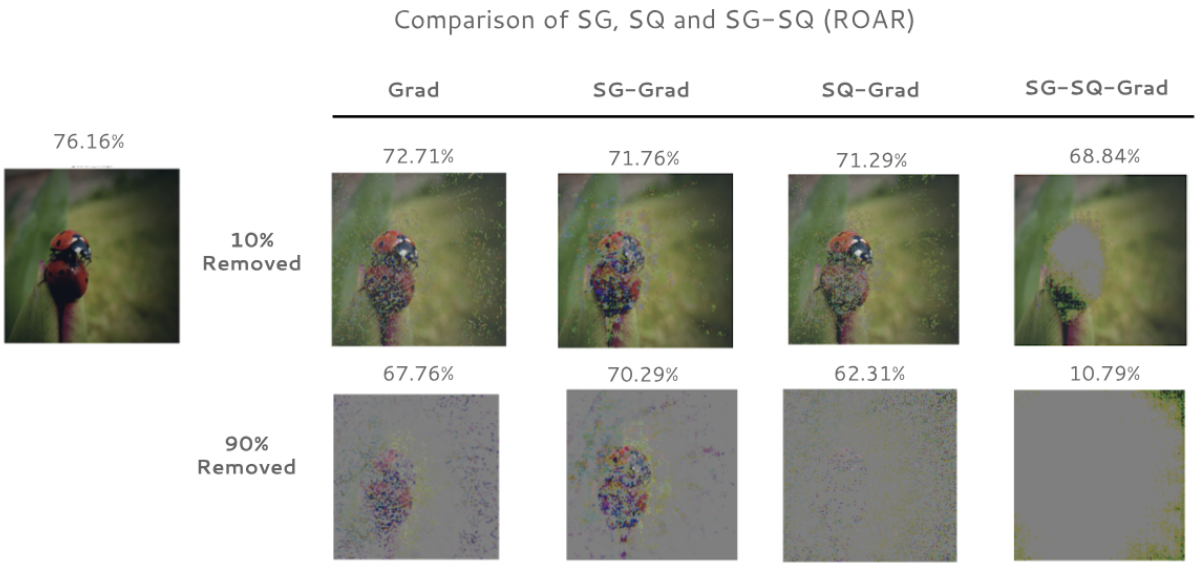

Figure 3. Certain transformations of the estimate can substantially improve accuracy of all estimators. SG-SQ-GRAD far outperforms both SQ-GRAD and SG-GRAD. While squaring alone provides small gains to the accuracy of all estimators, only averaging noisy estimates is comparable in performance with the accuracy of a single gradient heatmap. **From left to right**: raw gradient heatmap (GRAD), average of noisy gradient heatmaps (SG-GRAD), squared gradient heatmap (SG-GRAD) and both squaring a set of noisy gradient heatmaps and then averaging (SG-SQ-GRAD). Average test-set accuracy across three independent training runs is displayed above each modification variant.

$\mathbf{x}_0$ to the actual input $\mathbf{x}$. All of these estimates are then aggregated together:

$$\mathbf{e} = (\mathbf{x} - \mathbf{x}_0) \odot \int_{\alpha=0}^{1} \frac{\partial f_w(\mathbf{x}_0 + \alpha(\mathbf{x} - \mathbf{x}_0))}{\partial \mathbf{x_i}}.$$

The reference point should reflect the absence of the input. In this work, we use a black image as the reference point as suggested by (Sundararajan et al., 2017). Additional examples of attribution methods include Layerwise-Relevance Propagation (Bach et al., 2015), the Deep-Taylor Decomposition (Montavon et al., 2017), DeepLift (Shrikumar et al., 2016) and PatternAttribution (Kindermans et al., 2017). We again note that the performance of IG should not be considered representative of other attribution methods and that evaluating the accuracy of additional attribution estimators should be the subject of future work.

SMOOTHGRAD (SG), SQUARED (SQ) and SMOOTHGRAD-SQUARED (SG-SQ) Finally, we also evaluate SG (Smilkov et al., 2017) which proposes averaging T noisy estimates of feature importance (for any method). The set of noisy estimates are constructed by injecting a single input with Gaussian noise η independently T times to construct a set of noisy inputs. The estimate of feature importance becomes:

$$\mathbf{e} = \sum_{i=1}^{T} (g_i(\mathbf{x} + \eta, A_n^l))$$

Although this was not described in the original publication, the default open-source implementation of SG squares each estimate $\mathbf{e}$ before averaging the estimates:

$$\mathbf{e} = \sum_{l=1}^{P} (g_i(\mathbf{x} + \eta, A_n^l)^2)$$

We evaluate both methods; we refer to **both** averaging and squaring estimates as SG-SQ, **only averaging** as SG and **only squaring** as SQ.

Random binary assignment of importance Finally we consider a baseline that assigns a random ranking to the input features. g^R assigns a random binary importance probability $\mathbf{e} \mapsto 0, 1$. This amounts to a binary vector $\mathbf{e} \sim Bernoulli(1-t)$ where $(1-t)$ is the probability of $e_i = 1$.

3.3. Training a new model is critical

The replacement value c can only be considered uninformative if the value is a variable present in the distribution X but irrelevant to the classification task: $c \perp\!\!\!\perp Y$. **Training a new model is crucial** so that f_w^M, where w specifies the model weights, is trained on a distribution X that includes c, since it is only in this case that the model can learn that c is an uninformative value. By including c in the input distribution, the estimated $p_w(y, x^M)$ can approximate the

true distribution of $p(y, x^M)$. In the case without retraining, $f_w(x)$ has been trained on a distribution $p(y, x)$ but is expected to approximate $p(y, x^M)$ at inference time.

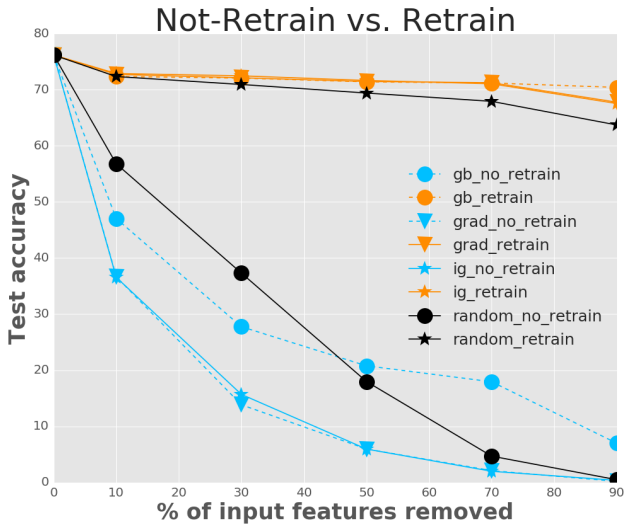

Figure 4. Comparison of accuracy degradation between a model not retrained on the modified inputs (NO-RETRAIN) and a model that is RETRAINED on the modified inputs. Accuracy test curves for datasets with a varying fraction of all inputs modified according to the rankings of the following estimators: Guided Backprop (GB-RETRAIN and GB-NO-RETRAIN), Integrated Gradients (IG-RETRAIN and IG-NO-RETRAIN) and Grad (GRAD-RETRAIN and GRAD-NO-RETRAIN). A model that is not retrained presents far higher accuracy degradation for all modification thresholds. The noticeable gap in degradation suggests that the measuring performance on a NO-RETRAIN model may capture confounding degradation caused by the introduction of artifacts that lie outside of the trained manifold.

4. Experimental Framework and Results

4.1. Experiment Framework

We use the ResNet-50 model model for both generating feature importance estimates and training. ResNet-50 was chosen because of the public code implementations (in both PyTorch (Gross & Wilber, 2017) and Tensorflow (Abadi et al., 2015)) and because it can be trained to give near to state of art performance in a reasonable amount of time (Goyal et al., 2017).

For all 1.28 million train and $50,000$ validation images in ImageNet we first apply test time pre-processing as used by Goyal et al. (2017). We compute an estimate $\mathbf{e}$ for every input in the training and test set. For all estimators, A_n^l is pre-softmax activation for the model prediction. We rank each $\mathbf{e}$ into an ordered set $\{e_i^o\}_{i=1}^N$. For the top t fraction of this ordered set, we replace the corresponding pixels in the raw image with the per channel mean of a random sample of $2,000$ images in ImageNet.Fig. 2 shows an example image from each modified ImageNet dataset.

We train 15 models in total per estimator, with 3 separate runs for each fraction $t = [0.1, 0.3, 0.5, 0.7, 0.9]$. We report test accuracy as the average of these 3 runs. Analogous to (Goyal et al., 2017), we train each model for 90 epochs with linear learning rate warmup for the first 5 epochs. A minor modification is that we use a batch size of 4096 and a base learning rate of 4.8. We also use a ghost batch norm size of 128 instead of 32. In our base implementation, the ResNet-50 trained on an unmodified ImageNet dataset achieves a mean accuracy of $76.16\% \pm 0.15$ (averaged over 5 runs). This is comparable to the performance reported by (Goyal et al., 2017).

4.2. Experiment Results

Experiment Variants In addition to ROAR, **R**em**O**ve **A**nd **R**etrain, we evaluate the opposite approach of **KAR**, **K**eep **A**nd **R**etrain. While ROAR removes features by replacing a fraction of inputs estimated to be most important, KAR preserves the inputs considered to be most important.

4.2.1. MODEL PERFORMANCE IS REMARKABLY ROBUST TO RANDOM MODIFICATION

The estimator g^R assigns importance at random to all inputs. Comparing estimators to this baseline allows us to answer the question: is the estimate of importance more accurate than a random guess? The performance of the random baseline is surprising. After replacing a large portion of all inputs with a constant value, the model not only trains but it still performs surprisingly well. In the appendix we show an example image from each randomly modified dataset and the associated test-set performance. When we replace 90% of inputs with a constant uninformative value, we are still able to achieve a test accuracy of 63.71%. In fact, we find the model still achieves 45.40% despite removing 99% of features .

The ability of the model to extract a meaningful representation from a small random fraction of inputs suggests a case where many inputs are likely redundant. The nature of our input—an image where correlations between pixels are expected—provides one possible reason for this redundancy.

4.2.2. KAR: PRESERVING FEATURES IS A LESS DISCRIMINATIVE BENCHMARK THAN ROAR

KAR, **K**eep **A**nd **R**etrain, **preserves** rather than replaces a fraction all input features estimated to be most important. All other features are modified by replacing the raw input feature with a constant mean value. In the appendix, we visualize a single image modified according to KAR while

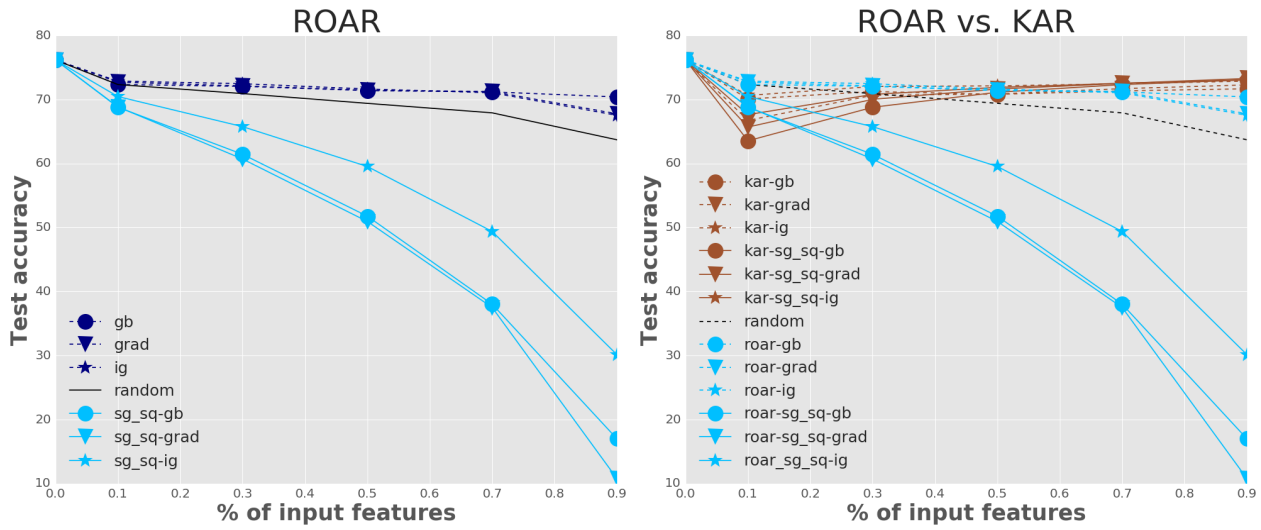

Figure 5. **Left inset:** ROAR measures accuracy by removing features. Without applying SG-SQ all estimators considered (GB,GRAD, IG) perform worse than a random modification of the input. However, when a set of noisy estimates are squared and averaged (SG-SQ-GB, SG-SQ-IG and SG-SQ-GRAD) we see a sharp degradation in performance and all estimators far outperform a random guess. **Right inset:** Comparison of KAR vs. ROAR. Inputs modified according to KAR result in a very narrow range of model accuracy. This suggests that KAR is an easier benchmark to fulfill, and does not produce the divergence in performance visible in ROAR.

varying the fraction of input features modified. Since we keep the important information rather than remove it, a high test accuracy relative to all other methods is desirable when evaluating KAR. In the right inset chart of Fig. 5 we plot KAR on the same curve as ROAR to enable a more intuitive comparison between the benchmarks. The comparison suggests that KAR appears to be a poor discriminator between estimators. The x-axis indicates the fraction of features that are preserved/removed for KAR/ROAR respectively.

Under KAR, all estimators and the SG-SQ transformation of estimators perform in a similar range to each other. We expect to observe the largest divergence in performance between estimators when $t = 0.1$ and the lowest ranked 90% of input features are modified. At the point of greatest divergence ($t = .1$) there is only a 7.08% accuracy difference between the highest (IG) and lowest (SG-SQ-GB) ranking estimator. In comparison, ROAR, at the point of greatest divergence ($t = .9$) presents a 59.6% difference between the highest (SG-SQ-GRAD) and lowest (GB) ranking estimator.

These findings suggest that the task of identifying features to preserve is an easier benchmark to fulfill than accurately identifying a fraction of input that will cause the maximum damage to the model performance. Under KAR, no estimator consistently proves the most accurate across all fractions of t, although GB and SG-SQ-GB are consistently ranked the least accurate for all thresholds.

4.2.3. ROAR: SG-SQ FAR OUTPERFORMS THE ACCURACY OF OTHER ESTIMATORS

ROAR removes the input features estimated to be most important. In the appendix, we show an example of how the dataset is modified according to ROAR for all fractions (t) that we evaluate. Under ROAR, the most desirable estimator causes the most model accuracy degradation (the lowest accuracy). The left inset of Fig. 5 shows how the small subset of estimators that we consider (GB, IG, GRAD) degrades accuracy less than a random ranking of input features for all thresholds t. As a larger fraction of the inputs are modified, the gap in performance between a random ranking and all other estimators grows. We expect to see the largest divergence between estimator performance to occur when $t = .9$, and we have replaced the top 90% of all input feature. When $t = .9$, the gap between a random ranking and the nearest estimator in accuracy grows from 0.01% (when $t = .1$) to 3.81%.

The estimators GB, IG, GRAD do not show a wide range in performance unless certain transformations (like averaging a set of squared, noisy estimates) are applied. Without this transformation, even at $t = .9$ there is little difference in accuracy between IG, GRAD at 67.52% and 67.76% respectively. At lower fractions of t, GB was the most accurate estimator with narrow leads; however, at high fractions of t, GB is the least accurate estimator out of all those considered.

When we square and average estimates (SG-SQ) all es-

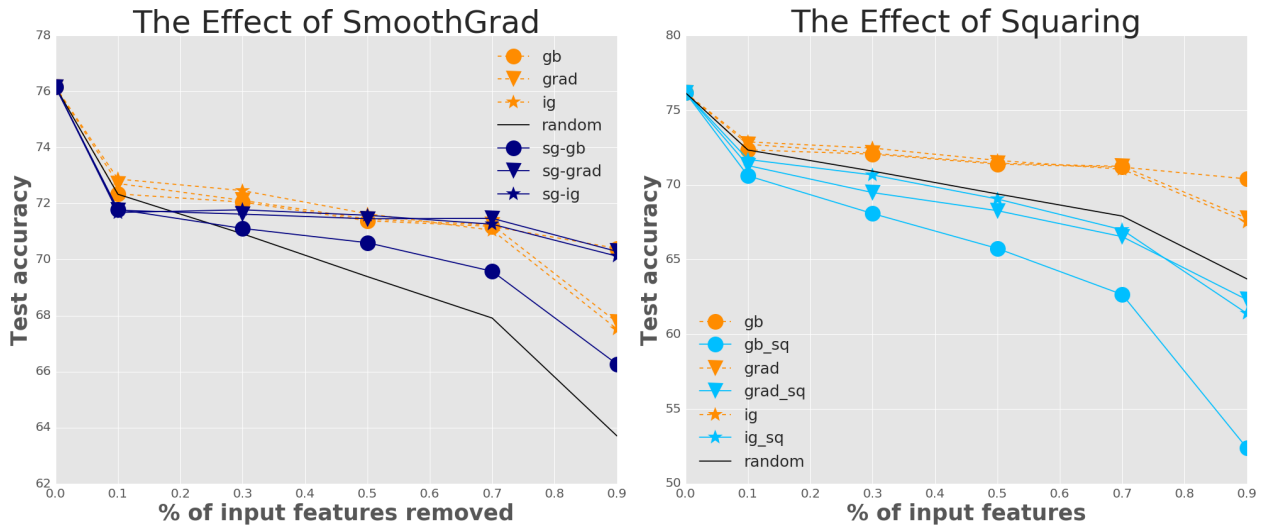

Figure 6. **Left inset:** SG, which only averages a set of noisy estimates performs worse than a random assignment of importance and is on par with a single estimate of importance. **Right inset:** All estimators (GB, IG, GRAD) benefit from squaring the estimate before ranking. A squared ranking of estimates improves performance from slightly less than a random baseline to slightly better than such a random guess.

timators substantially benefit and far outperform such a random guess. At $t = .9$, SG-SQ-GRAD is the most accurate estimator with a test-set accuracy o $10.78\% \pm 7.85$. SG-SQ-GB is also very effective with a final test accuracy of $17\% \pm 3.45$. The surprising effects of SG-SQ deserve further investigation; why is averaging a set of squared noisy estimates so effective at improving the accuracy of the ranking? To disentangle whether both squaring and then averaging are required, we explore whether we achieve similar performance gains by **only** squaring estimates SQ or by **only** averaging a set of noisy estimates SG.Fig. 3 shows the modification for a single image and the accuracy according to each of these variants.

4.2.4. ROAR: ONLY AVERAGING NOISY ESTIMATES DOES NOT PRODUCE GAINS IN PERFORMANCE

We benchmark the *original* SG formulation by Smilkov et al. (2017) which averages a set of 15 noisy estimates *without* squaring. Our results in the left inset chart of Fig. 6 are intriguing. Simply averaging noisy estimates appears no better than a single estimate for the subset of all estimators that we evaluate. For example, at $t = 0.9$ SG-GRAD and SG-IG are surprisingly less accurate than a single estimate by 2.52% and 2.6% respectively. Only GB appears to benefit at every threshold t from averaging; SG-GB improves accuracy relative to GB by 4.13% when $t = .9$. Despite this small gain, for all estimators (GB, IG, GRAD) averaging a set of noisy estimates remains less accurate than a random ranking for almost all values of t.

4.2.5. ROAR: ONLY SQUARING PROVIDES A SMALL GAIN IN ACCURACY

Squaring benefits the accuracy of all estimators that we considered. In the right inset chart of Fig. 6, we can see that squared estimates perform better than the raw estimate. When squared, an estimate gains slightly more accuracy than a random ranking of input features. In particular, squaring benefits GB; at $t = .9$ performance of SQ-GB relative to GB improves by $8.43\% \pm 0.97$.

Squaring is an equivalent transformation to taking the absolute value of the estimate before ranking all inputs. After squaring, negative estimates become positive, and the ranking then only depends upon the magnitude and not the direction of the estimate. The benefits gained by squaring furthers our understanding of how the direction of GB, IG and GRAD values should be treated. For all these estimators, estimates are very much a reflection of the weights of the network. The magnitude may be far more telling of feature importance than direction; a negative signal may be just as important as positive contributions towards a model's prediction. While squaring improves the accuracy of all estimators, the transformation does not explain the observed large gains in accuracy that we observe when we average a set of noisy squared estimates. For example, at $t = .9$ while SQ-GRAD improves the accuracy of GRAD by 5.45%; its accuracy pales against SG-SQ-GRAD which improves the accuracy of GRAD by 56.97%. This remains an important open research question that requires further treatment.

5. Conclusion and Future Work

In this work, we propose ROAR to evaluate the relative accuracy of input feature importance estimators. We show that training is crucial if the replacement value is to be considered uninformative, while demonstrating how ROAR is a harder benchmark to satisfy than KAR. We also find that the subset of estimators that we evaluate using ROAR degrade performance less than a random ranking of feature importance. However, certain transformations (squaring estimates SQ and squaring combined with averaging noisy estimates SG-SQ) of the estimate benefit the accuracy of all estimators. There are certain limitations to this evaluation framework: we consider a single model, test on a single (albeit large-scale) dataset, and due to computational constraints we only consider a small subset of existing estimators for the initial benchmark. While the current study is focused on evaluating a core set of approaches, we invite future collaborations to assess the relative accuracy of more methods.

ACKNOWLEDGMENTS

We thank Douglas Eck, Jonas Kemp, Melissa Fabros, Julius Abedayo, Simon Kornblith, Niru Maheswaranathan, Gamaleldin Elsayed and Andrew Ross for their thoughtful feedback on earlier iterations of this work.

References

Abadi, Martín, Agarwal, Ashish, Barham, Paul, Brevdo, Eugene, Chen, Zhifeng, Citro, Craig, Corrado, Greg S., Davis, Andy, Dean, Jeffrey, Devin, Matthieu, Ghemawat, Sanjay, Goodfellow, Ian, Harp, Andrew, Irving, Geoffrey, Isard, Michael, Jia, Yangqing, Jozefowicz, Rafal, Kaiser, Lukasz, Kudlur, Manjunath, Levenberg, Josh, Mané, Dandelion, Monga, Rajat, Moore, Sherry, Murray, Derek, Olah, Chris, Schuster, Mike, Shlens, Jonathon, Steiner, Benoit, Sutskever, Ilya, Talwar, Kunal, Tucker, Paul, Vanhoucke, Vincent, Vasudevan, Vijay, Viégas, Fernanda, Vinyals, Oriol, Warden, Pete, Wattenberg, Martin, Wicke, Martin, Yu, Yuan, and Zheng, Xiaoqiang. TensorFlow: Large-scale machine learning on heterogeneous systems, January 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

Adebayo, Julius, Gilmer, Justin, Goodfellow, Ian, and Kim, Been. Local explanation methods for deep neural networks lack sensitivity to parameter values. *ICLR Workshop*, 2018.

Ancona, M., Ceolini, E., Öztireli, C., and Gross, M. Towards better understanding of gradient-based attribution methods for Deep Neural Networks. *ArXiv e-prints*, November 2017.

Ba, Lei Jimmy and Caruana, Rich. Do deep nets really need to be deep? In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pp. 2654–2662, Cambridge, MA, USA, 2014. MIT Press. URL http://dl.acm.org/citation.cfm?id=2969033.2969123.

Bach, Sebastian, Binder, Alexander, Montavon, Grégoire, Klauschen, Frederick, Müller, Klaus-Robert, and Samek, Wojciech. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.

Baehrens, David, Schroeter, Timon, Harmeling, Stefan, Kawanabe, Motoaki, Hansen, Katja, and Müller, Klaus-Robert. How to explain individual classification decisions. *Journal of Machine Learning Research*, 11(Jun): 1803–1831, 2010.

Dabkowski, P. and Gal, Y. Real Time Image Saliency for Black Box Classifiers. *ArXiv e-prints*, May 2017.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

Fong, Ruth C. and Vedaldi, Andrea. Interpretable explanations of black boxes by meaningful perturbation. In *ICCV*, pp. 3449–3457. IEEE Computer Society, 2017.

Frosst, N. and Hinton, G. Distilling a Neural Network Into a Soft Decision Tree. *ArXiv e-prints*, November 2017.

Goyal, P., Dollár, P., Girshick, R., Noordhuis, P., Wesolowski, L., Kyrola, A., Tulloch, A., Jia, Y., and He, K. Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour. *ArXiv e-prints*, June 2017.

Gross, S. and Wilber, M. Training and investigating Residual Nets. https://github.com/facebook/fb.resnet.torch, January 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning for Image Recognition. *ArXiv e-prints*, December 2015.

Kim, B., Wattenberg, M., Gilmer, J., Cai, C., Wexler, J., Viegas, F., and Sayres, R. Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV). *ArXiv e-prints*, November 2017.

Kindermans, P.-J., Hooker, S., Adebayo, J., Alber, M., Schütt, K. T., Dähne, S., Erhan, D., and Kim, B. The (Un)reliability of saliency methods. *ArXiv e-prints*, November 2017.

Kindermans, Pieter-Jan, Schütt, Kristof T, Alber, Maximilian, Müller, Klaus-Robert, Erhan, Dumitru, Kim, Been,

and Dähne, Sven. Learning how to explain neural networks: Patternnet and patternattribution. *arXiv preprint arXiv:1705.05598v2*, 2017.

Koh, Pang Wei and Liang, Percy. Understanding black-box predictions via influence functions. In Precup, Doina and Teh, Yee Whye (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1885–1894, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

Montavon, Grégoire, Lapuschkin, Sebastian, Binder, Alexander, Samek, Wojciech, and Müller, Klaus-Robert. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, 65:211–222, 2017.

Morcos, A. S., Barrett, D. G. T., Rabinowitz, N. C., and Botvinick, M. On the importance of single directions for generalization. *ArXiv e-prints*, March 2018.

Olah, Chris, Mordvintsev, Alexander, and Schubert, Ludwig. Feature visualization. *Distill*, 2017. doi: 10.23915/distill.00007. https://distill.pub/2017/feature-visualization.

Raghu, Maithra, Gilmer, Justin, Yosinski, Jason, and Sohl-Dickstein, Jascha. SVCCA: singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *NIPS*, pp. 6078–6087, 2017.

Ross, Andrew Slavin and Doshi-Velez, Finale. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. *CoRR*, abs/1711.09404, 2017.

Ross, Andrew Slavin, Hughes, Michael C., and Doshi-Velez, Finale. Right for the right reasons: Training differentiable models by constraining their explanations. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pp. 2662–2670, 2017.

Samek, W., Binder, A., Montavon, G., Lapuschkin, S., and Mller, K. R. Evaluating the Visualization of What a Deep Neural Network Has Learned. *IEEE Transactions on Neural Networks and Learning Systems*, 28(11):2660–2673, Nov 2017. ISSN 2162-237X. doi: 10.1109/TNNLS.2016.2599820.

Selvaraju, Ramprasaath R., Cogswell, Michael, Das, Abhishek, Vedantam, Ramakrishna, Parikh, Devi, and Batra, Dhruv. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

Shrikumar, A., Greenside, P., Shcherbina, A., and Kundaje, A. Not Just a Black Box: Learning Important Features Through Propagating Activation Differences. *ArXiv e-prints*, May 2016.

Shrikumar, A., Greenside, P., and Kundaje, A. Learning Important Features Through Propagating Activation Differences. *ArXiv e-prints*, April 2017.

Simonyan, Karen and Zisserman, Andrew. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

Smilkov, Daniel, Thorat, Nikhil, Kim, Been, Viégas, Fernanda, and Wattenberg, Martin. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

Springenberg, Jost Tobias, Dosovitskiy, Alexey, Brox, Thomas, and Riedmiller, Martin. Striving for simplicity: The all convolutional net. In *ICLR*, 2015.

Sundararajan, Mukund, Taly, Ankur, and Yan, Qiqi. Axiomatic attribution for deep networks. *arXiv preprint arXiv:1703.01365*, 2017.

van der Maaten, Laurens and Hinton, Geoffrey. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008. URL http://www.jmlr.org/papers/v9/vandermaaten08a.html.

Wu, M., Hughes, M. C., Parbhoo, S., Zazzi, M., Roth, V., and Doshi-Velez, F. Beyond Sparsity: Tree Regularization of Deep Models for Interpretability. *ArXiv e-prints*, November 2017.

Zeiler, Matthew D and Fergus, Rob. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pp. 818–833. Springer, 2014.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. *ArXiv e-prints*, November 2016a.

Zhang, J., Lin, Z., Brandt, J., Shen, X., and Sclaroff, S. Top-down Neural Attention by Excitation Backprop. *ArXiv e-prints*, August 2016b.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. Object Detectors Emerge in Deep Scene CNNs. *ArXiv e-prints*, 2014.

Zintgraf, Luisa M, Cohen, Taco S, Adel, Tameem, and Welling, Max. Visualizing deep neural network decisions: Prediction difference analysis. In *ICLR*, 2017.

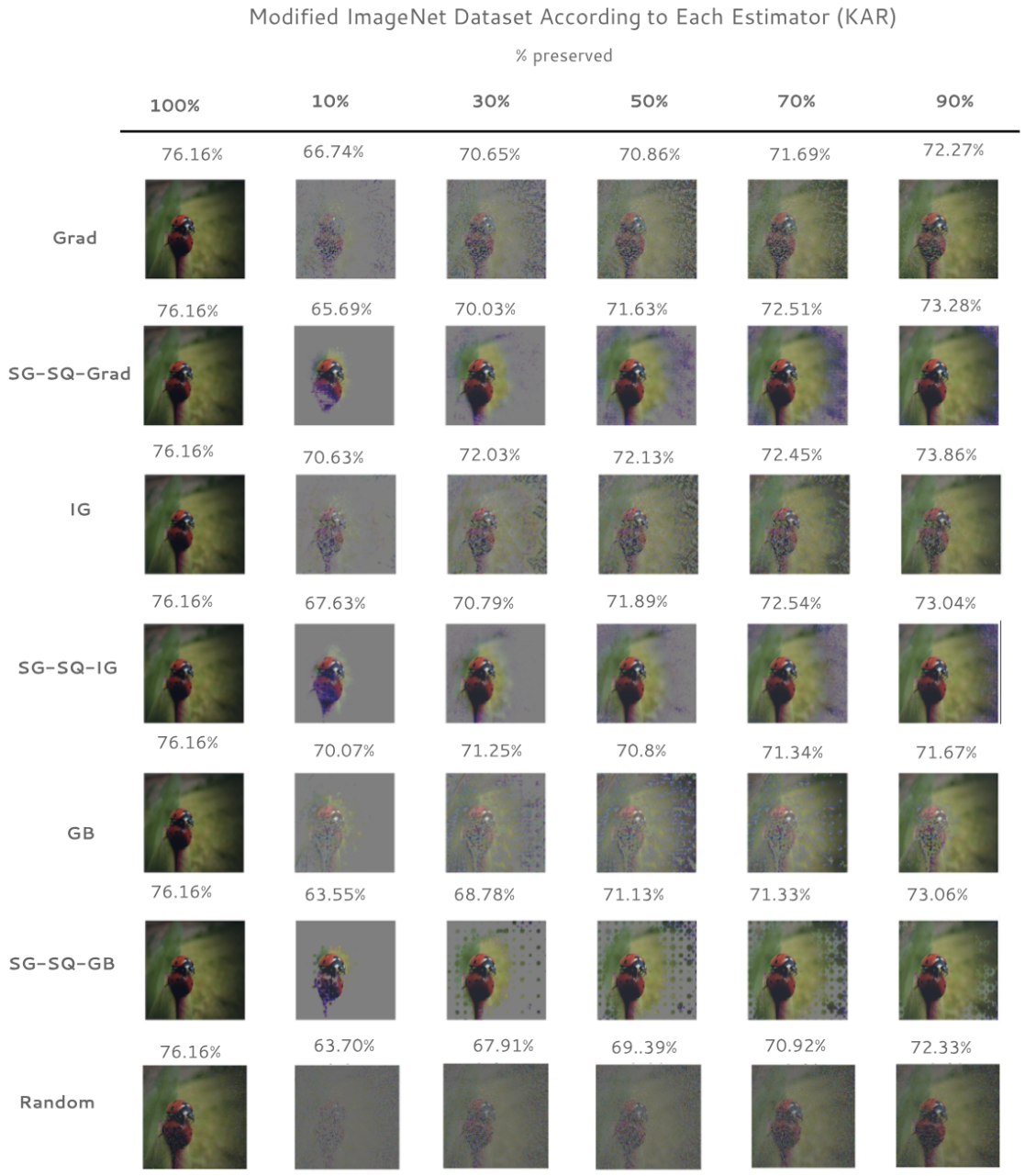

Figure 7. For **KAR**, **K**eep **A**nd **R**etrain, we keep a fraction of features estimated to be most important and replace the remaining features with a constant mean value. The most accurate estimator is the one that preserves model performance the most for a given fraction of inputs removed (the highest test-set accuracy). **From top to bottom**: gradient heatmap (GRAD), SmoothGrad-Squared Grad (SG-SQ-GRAD), Guided BackProp (GB), SmoothGrad-Squared Guided BackProp (SG-SQ-GB), Integrated Gradients IG SmoothGrad-Squared Integrated Gradients (SG-SQ-IG). The average test-set accuracy across three independent runs associated with each modification is displayed above each image.

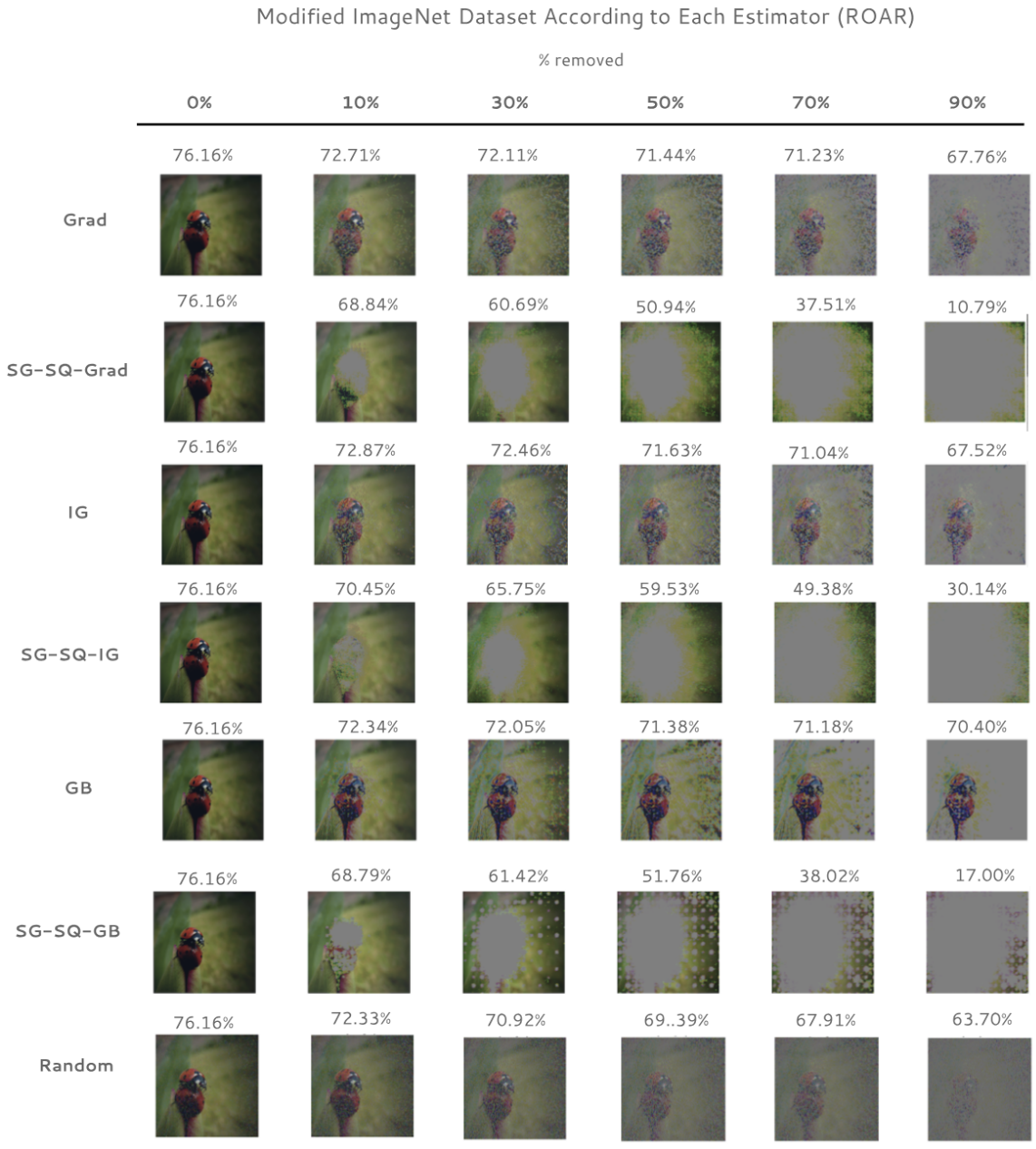

Figure 8. For **ROAR**, **R**em**0**ve **A**nd **Retrain** we remove features by replacing a fraction of the inputs estimated to be most important according to each estimator with a constant mean value. The most accurate estimator is the one that degrades model performance the most for a given fraction of inputs removed. **From top to bottom**: gradient heatmap (GRAD), SmoothGrad-Squared Grad (SG-SQ-GRAD), Guided BackProp (GB), SmoothGrad-Squared Guided BackProp (SG-SQ-GB), Integrated Gradients (IG) and SmoothGrad-Squared Integrated Gradients (SG-SQ-IG). The average test-set accuracy across three independent runs associated with each modification is displayed above each image.