Learning Deep Features for Discriminative Localization
[Paper] | [Notes] | [Implementation]

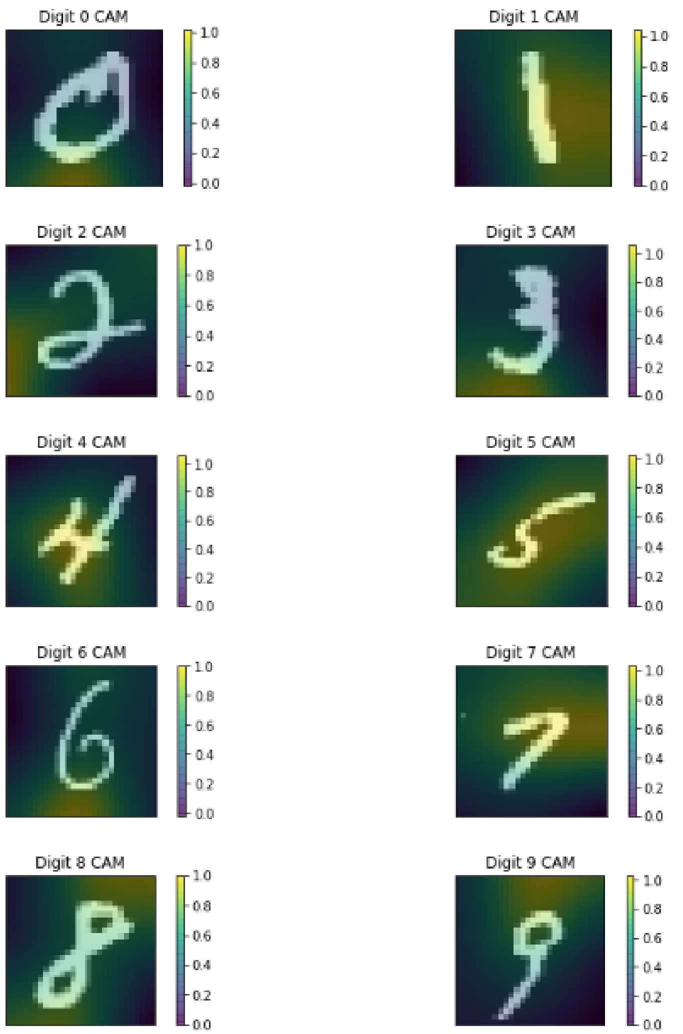
Class Activation Map (CAM) refers to the weighted activation maps generated for a given image. The best results for CAM are on CNNs which consist of blocks of CONV layers followed by a single AVGPOOL layer. The output from the AVGPOOL layer is then fed into a single DENSE layer which acts as a classifier.

$$M_c(x, y) = \sum_k w_k^c f_k(x, y).$$

Formula for calculating Class Activation Maps for a given class c.

CAM is computed by multiplying the activation maps from the very last CONV layer by the weights of the DENSE classification layer for the chosen class. The low-resolution map is then upsampled to the size of the input image. The input image and the generated CAM are then being shown together to showcase the parts of the image which have had the highest importance for the classified class.
Furthermore, CAM can be used as a localization tool with promising results.

Results



Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization

Paper | Notes | Implementation

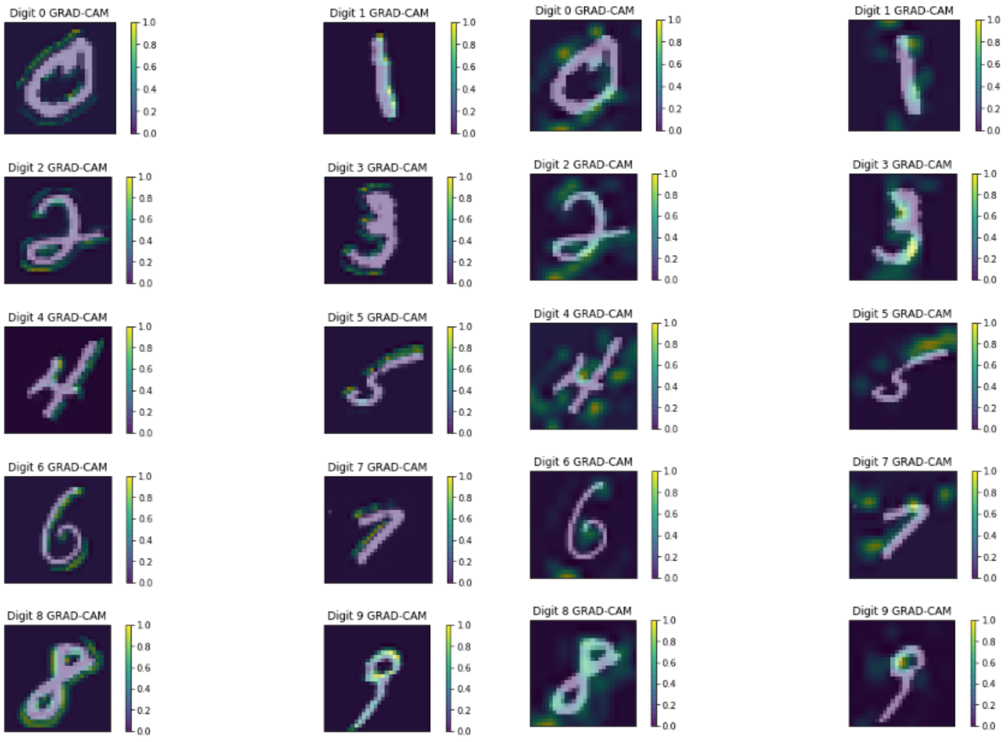
Gradient-Weighted Class Activation Maps (Grad-CAM) is a generalization of the CAM method, which uses the gradient signal instead of the weights of the last layer for weighing the activations. This makes the method reusable for any kind of CNN models unlike the original CAM method. Furthermore, Grad-CAM can be applied to practically any layer of the CNN model which produces a meaningful gradient signal.

$$L^c_{\text{Grad-CAM}} = ReLU\underbrace{\left(\sum_k \alpha_k^c A^k\right)}_{\text{linear combination}}$$

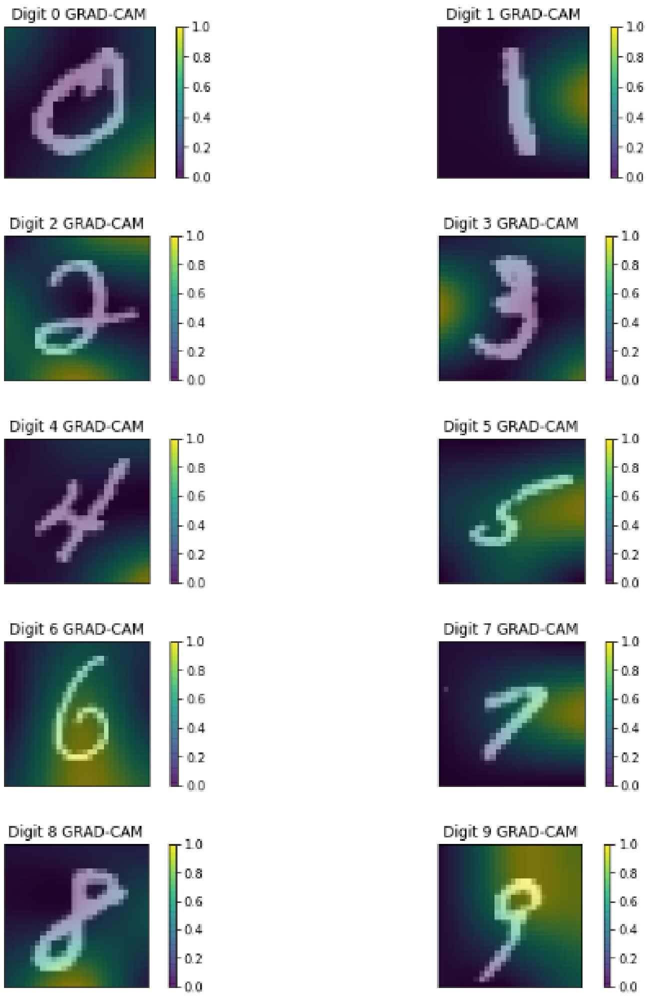
Formula for calculating Grad-CAM for a given class c.

Grad-CAM is being computed by multiplying the activations from the forward pass of the chosen layers with global-averaged-pooled incoming gradient from the backward pass. The result of the multiplication is then run through a ReLU activation. The final result is upsampled to the dimensions of the original input.
Similarly to CAM, Grad-CAM can be used as a localization tool with promising results. Unlike CAM, Grad-CAM provides good results for a wide variety of CNN model-families, without the need for architectural changes or auxiliary training.

Results for first (left) and second (right) CONV layers



Results for third (final) CONV layer

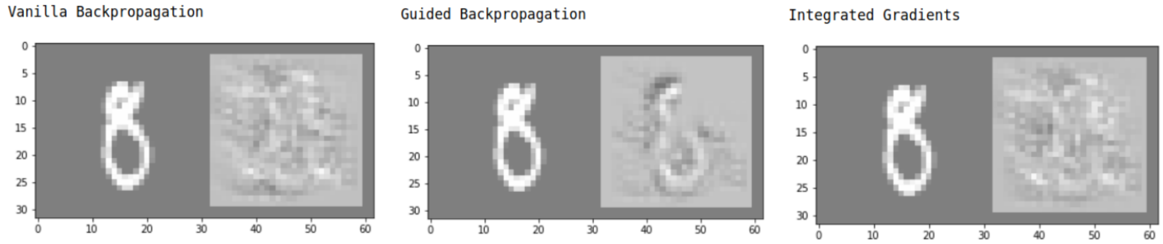

Digit 0 GRAD-CAM
Digit 1 GRAD-CAM
Digit 2 GRAD-CAM
Digit 3 GRAD-CAM
Digit 4 GRAD-CAM
Digit 5 GRAD-CAM
Digit 6 GRAD-CAM
Digit 7 GRAD-CAM
Digit 8 GRAD-CAM
Digit 9 GRAD-CAM

Axiomatic Attribution for Deep Networks | Paper | Notes | Implementation

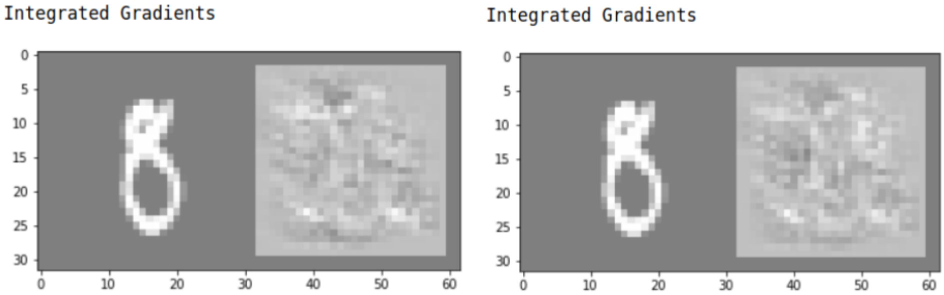
Integrated Gradients

Consider a **straightline** path in R^n from the baseline x'(black image) to the input x. **Integrated gradients** are defined as the path integral of the gradients along the straight line path from the vaseline x' to the input x. In practice we can construct a sequence of images interpolating from a baseline to the actual input image. Compute the gradient across these images in a loop and average these gradients to get the integrated gradients map. A formal definition is as follows

$$\text{IntegratedGrads}_i(x) ::= (x_i - x'_i) \times \int_{\alpha=0}^{1} \frac{\partial F(x' + \alpha \times (x - x'))}{\partial x_i} \, d\alpha$$



Comparison on vanilla backpropagation, guided backpropagation, integrated gradients. Integrated gradient map is qualitatively similar to the vanilla backpropagation map.



Comparison on different steps. In the left one the path is divided into **2** steps while the right one is **10000** steps. The result doesn't benefit from the amount of steps much.

Properties of Integrated Gradient

- **Sensitivity**: If for every input and baseline that differ in one feature but have different predictions then the differing feature should be given a non-zero attribution. Vanilla backpropagation, guided backpropagation, deconvnet, LRP, DeepLift violate this rule.
- **Implementation Invariance**: If outputs of two networks are equal for all inputs, despite having different implementation, they are functionally equivalent. DeepLift and LRP violate thie rule.
- Conservation (Completeness): Total attributions add up to the difference between the output of network at the input and at the baseline.

Layer-Wise Relevance Propagation: An Overview | Paper | Notes | Implementation

---

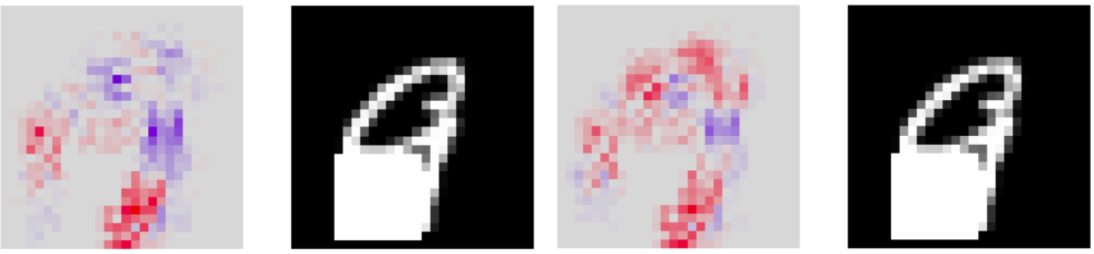
Comparison of different LRP rules

- **LRP-0:** The basic rule of LRP. However, it can be shown that a uniform application of this rule to the whole neural network provides an explanation that is equivalent to **Gradient×Input**. The result heatmap seems noisy, therefore one needs to design more robust propagation rules.

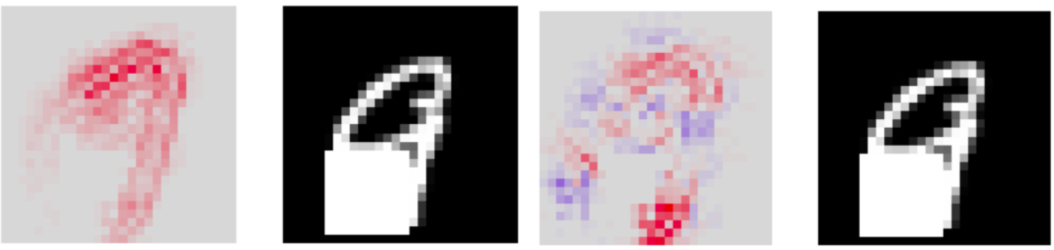
- **LRP-epsilon:** This enhancement of LRP is from adding a small positive term in the denominator. Epsilon is to absorb some relevance when the contributions to the activation of neuron is to weak or are weak or contradictory. Bigger epsilon leads to a **sparser heatmap** and **less noisy**.

- **LRP-gamma:** By favoring the effect of **positive contributions** over negative contributions we obtain this rule. As gamma increases, negative contributions start to disappear. The prevalence of positive contributions can limit how much positive and negative relevance can grow in LRP backpropagation, which leads to a more interpretable manner. Note when gamma is close to infinity large, LRP-gamma is equivalent to **LRP-a1b0**.

The following is a MNIST image nine but due to the white patch **predicted as 0** by our network.



Results from unifrom LRP-0(left) and uniform LRP-epsilon(right).

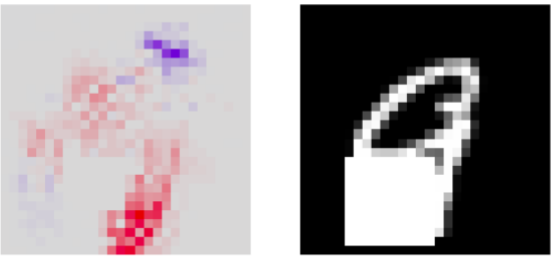


Results from uniform LRP-a1b0(left) and uniform LRP-a2b1(right) . For LRP-a1b0(left) it only indicates which features influence our network the most, but doesn't show negative relevance scores.

Which rule to choose?

Uniform LRP-0 heatmap is often overly complex and does not focus on salient features. The explanation is neither faithful nor understandable. Uniform LRP-epsilon heatmap keeps only a limited number features. It's a faithful explanation, but sometimes too sparse to be understandable. Uniform LRP-gamma (LRPa1b0) heatmap produce features that are more densely highlighted, but it also picks unrelated concepts and concepts that produce negative relevance, making it **unfaithful**.

A suggestion from this paper is pick a composite LRP where we use the **upper layers for LRP-0, middle layers for LRP-epsilon, and lower layers for LRP-gamma(LRP-a1b0).**



Results from Composite LRP

Name	Formula	Usage	DTD
LRP-0 [7]	$R_j = \sum_k \frac{a_j w_{jk}}{\sum_{0,j} a_j w_{jk}} R_k$	upper layers	✓
LRP-ϵ [7]	$R_j = \sum_k \frac{a_j w_{jk}}{\epsilon + \sum_{0,j} a_j w_{jk}} R_k$	middle layers	✓
LRP-γ	$R_j = \sum_k \frac{a_j (w_{jk} + \gamma w_{jk}^+)}{\sum_{0,j} a_j (w_{jk} + \gamma w_{jk}^+)} R_k$	lower layers	✓
LRP-$\alpha\beta$ [7]	$R_j = \sum_k \left(\alpha \frac{(a_j w_{jk})^+}{\sum_{0,j} (a_j w_{jk})^+} - \beta \frac{(a_j w_{jk})^-}{\sum_{0,j} (a_j w_{jk})^-} \right) R_k$	lower layers	×*
flat [30]	$R_j = \sum_k \frac{1}{\sum_j 1} R_k$	lower layers	×
w^2-rule [36]	$R_i = \sum_j \frac{w_{ij}^2}{\sum_i w_{ij}^2} R_j$	first layer ($\mathbb{R}^d$)	✓
$z^{\mathcal{B}}$-rule [36]	$R_i = \sum_j \frac{x_i w_{ij} - l_i w_{ij}^+ - h_i w_{ij}^-}{\sum_i x_i w_{ij} - l_i w_{ij}^+ - h_i w_{ij}^-} R_j$	first layer (pixels)	✓