Attention Branch Network: Learning of Attention Mechanism for Visual Explanation

(2019)

Hiroshi Fukui et al

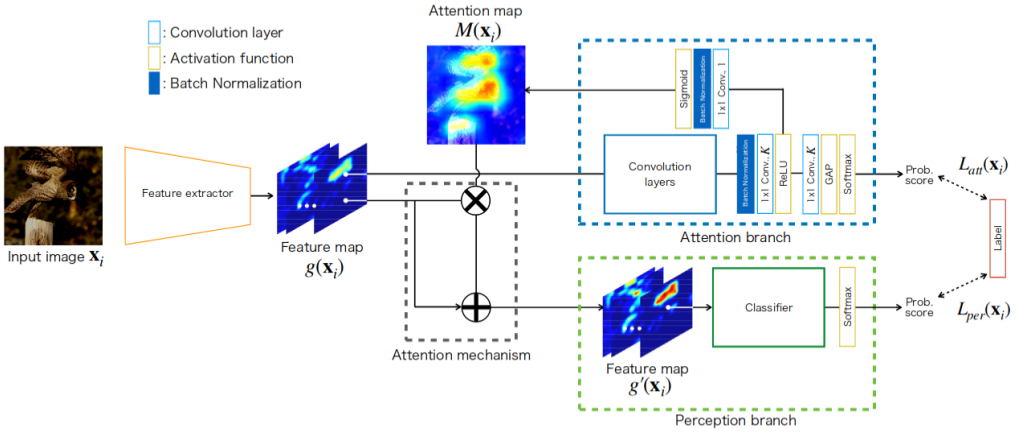
Notes

Intro

In trying to explain CNNs and interpret their predictions, various Visual Explanation methods have been proposed, either (1) gradients based: where we find the gradients of the weights with respect to a random noise (SmoothGrad), or the gradients of each class prediction with respect to the weights of a given features maps, generally the last layer before a fully connected layer or a global average pooling GAP (e.g GradCAM or GradCAM++). Or (2) response based, where we use the class scores after a GAP as weighting factors for the last feature maps, to obtain an attention map for each category based on its contribution to the classification scores.

The authors point out some problems with these approches, they are used only in inference, so any additionnal benefits of the attention maps that can increase the accuracy is discarded, for gradients methods, they're quite expensive requiring a backward pass for each class per image, and for CAM, replacing the FC layer with GAP migh reduce the performances, in this context they propose an attention branch network, where an additionnal branch is added to a classification network to produce the attention maps during training, these maps are applied to the classification branch and also used directly for classification, this yield better overall performances and detailed localization maps.

Attention Branch Network (ABN)



ABN is an extension to a classification network by adding a branch with the attention mecanism, the ABN consists of three components a fature extractor (the first three blocks in case of a ResNet), and attention branch (with a the fourth resnet block with out any strided convolution, followed by a set of conv1x1 & batch norms for generating the attention maps and classification scores) and a preception branch containing the rest of the resnet model (with an additionnal masking of its input using the attention maps)

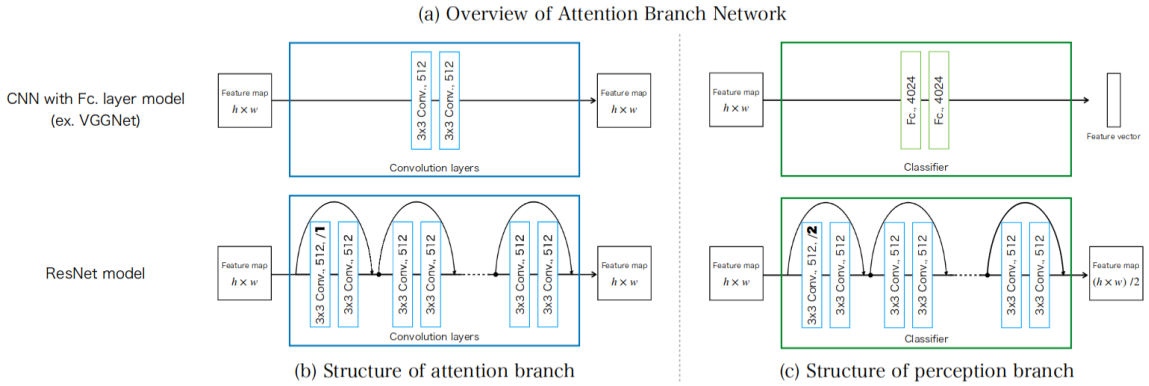


Figure 2. Detailed structure of Attention Branch Network.

Attention branch After the resnet block, a 1x1 conv with K output channels is applied to adjust the depth to the number of calsses, and then this feature maps of size $H \times W \times K$ where $H, W = input\_size/16$, is used in two manners, one to generated a single attention maps, using a $1x1x1conv$ resulting in a $H \times W \times 1$ attention map, and also passed through a GAP and then a softmax for classification scores.

In a multi-task setting, instead of outputing a single attention maps, given T tasks (like in CelebA dataset with 40 facial attributes, where we want to detect is the subject in the input iamage si similing, wearing a necklie, has blond hair etc.) the conv used for the attention map generation is of size $(1 \times 1 \times T)$ to get T attention maps.

Preception branch The preception branch is the rest of the network, the only difference is the new input $g'_c(x_i)$ is masked using the generated attention map $M(x_i)$ for input x_i, this is done as follows, so given the input to preception branch $g_c(x_i)$, the new input is $g'_c(x_i)$:

$$g'_c(x_i) = M(x_i)g_c(x_i)$$

$$g'_c(x_i) = (M(x_i) + 1)g_c(x_i)$$

In the experiments, the authors found the second way gives better results (1 + mask)* input.

In a multi-task setting, with T preception branches, and T generated attentation masks, we can apply each attention maps to the input of the corresponding preception branch.

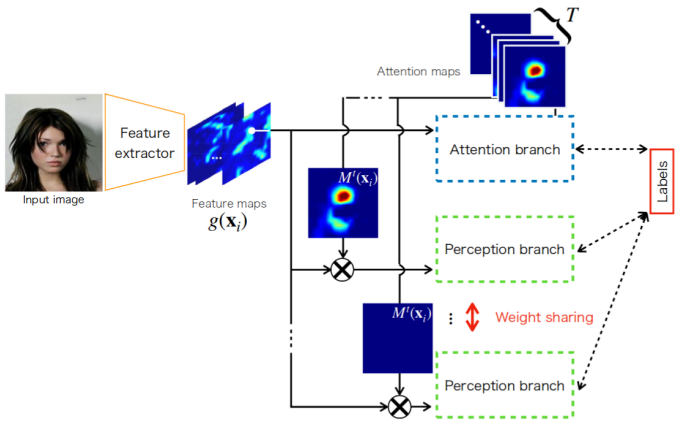


Figure 3. ABN for multi-task learning.

Experiments

Table 1. Comparison of the top-1 errors on CIFAR100 with attention mechanism.

	$g(\mathbf{x})$	$g(\mathbf{x}) \cdot M(\mathbf{x})$	$g(\mathbf{x}) \cdot (1 + M(\mathbf{x}))$
ResNet20	31.47	30.61	**30.46**
ResNet32	30.13	28.34	**27.91**
ResNet44	25.90	**24.83**	25.59
ResNet56	25.61	24.22	**24.07**
ResNet110	24.14	23.28	**22.82**

Table 2. Comparison of top-1 errors on CIFAR10, CIFAR100, SVHN, and ImageNet dataset.

Dataset	CIFAR10	CIFAR100	SVHN [23]	ImageNet [5]
VGGNet [14]	–	–	–	31.2
VGGNet+BN	–	–	–	26.24*
ResNet [9]	6.43	24.14*	2.18*	22.19*
VGGNet+CAM [41]	–	–	–	33.4
VGGNet+BN+CAM	–	–	–	27.42*$_{(+1.18)}$
ResNet+CAM	–	–	–	22.11*$_{(-0.08)}$
WideResNet [38]	4.00	19.25	2.42*	21.9
DenseNet [11]	4.51	22.27	2.07*	22.2
ResNeXt [34]	3.84*	18.32*	2.16*	22.4
Attention [32]	3.90	20.45	–	21.76
AttentionNeXt [32]	–	–	–	21.20
SENet [12]	–	–	–	21.57
VGGNet+BN+ABN	–	–	–	25.55 $_{(-0.69)}$
ResNet+ABN	4.91 $_{(-1.52)}$	22.82 $_{(-1.32)}$	**1.86** $_{(\mathbf{-0.32})}$	21.37 $_{(-0.82)}$
WideResNet+ABN	**3.78** $_{(\mathbf{-0.22})}$	18.12 $_{(-1.13)}$	2.24 $_{(-0.18)}$	–
DenseNet+ABN	4.17 $_{(-0.34)}$	21.63 $_{(-0.64)}$	2.01 $_{(-0.06)}$	–
ResNeXt+ABN	3.80 $_{(-0.04)}$	**17.70** $_{(\mathbf{-0.62})}$	2.01 $_{(-0.15)}$	–
SENet+ABN	–	–	–	**20.77** $_{(\mathbf{-0.80})}$

$*$ *indicates results of re-implementation accuracy*

task	model [%]	maker [%]
VGG16	85.9	90.4
ResNet101	90.2	90.1
VGG16+ABN	90.7	92.9
ResNet101+ABN	**97.1**	**98.1**