Top-GAP: Integrating Size Priors in CNNs for more Interpretability, Robustness, and Bias Mitigation

Lars Nieradzik[1], Henrike Stephani[1], and Janis Keuper[2]

[1] Image Processing, Fraunhofer ITWM, Fraunhofer Platz 1, Kaiserslautern, 67663, Germany
{lars.nieradzik,henrike.stephani}@itwm.fraunhofer.de
[2] Institute for Machine Learning and Analysis, Offenburg University, Badstr. 24, Offenburg, 77652, Germany
keuper@imla.ai

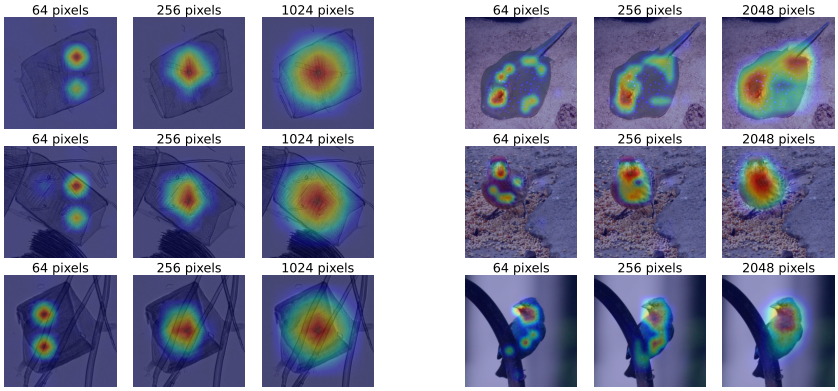
Abstract. This paper introduces Top-GAP, a novel regularization technique that enhances the explainability and robustness of convolutional neural networks. By constraining the spatial size of the learned feature representation, our method forces the network to focus on the most salient image regions, effectively reducing background influence. Using adversarial attacks and the Effective Receptive Field, we show that Top-GAP directs more attention towards object pixels rather than the background. This leads to enhanced interpretability and robustness. We achieve over 50% robust accuracy on CIFAR-10 with PGD $\epsilon = 8/255$ and 20 iterations while maintaining the original clean accuracy. Furthermore, we see increases of up to 5% accuracy against distribution shifts. Our approach also yields more precise object localization, as evidenced by up to 25% improvement in Intersection over Union (IOU) compared to methods like GradCAM and Recipro-CAM.

Keywords: Class activation maps · Robustness · Adversarial attacks

1 Introduction

Modern computer vision has made remarkable progress with the proliferation of Deep Learning, particularly convolutional neural networks (CNNs). These networks have demonstrated unprecedented capabilities in tasks ranging from image classification to semantic segmentation [54]. However, the explainability of these models remains a critical problem.

Many previous attempts to improve explainability have focused on improving class activation maps of the already trained networks. We propose a different approach that focuses on a novel method to regularize the network during training. A constraint is added to the training procedure that limits the spatial size of the learned feature representation which a neural network can use for a prediction. Unlike [35], we do not need KKT conditions or the Lagrangian. The disadvantage of direct constrained optimization is that it can make gradient descent fail to



(a) Wood identification [32] **(b)** ImageNet [11]

Fig. 1: Example images from a biological classification dataset (a) and ImageNet (b), where we limit the locations in the output feature map that the CNN can use to make predictions. Increasing the allowed pixel count leads to more pixels being highlighted in the class activation map (CAM). If the object size is not known or variable, the pixel constraint with the highest accuracy can be selected.

converge if the algorithm is not modified. Instead, we force the network to only use the most important k locations in the feature map. The "importance" stems from an additional sparsity loss that forces the network to output an empty feature map. Part of the loss tries to increase k locations, while another part tries to set all of them to zero. This constraint simplifies the optimization problem and allows us to keep the same accuracy as the unconstrained problem.

Restricting the output feature maps fundamentally changes the way the network works internally. In Fig. 1, we see an example on how the constraint affects the class activation map (CAM). We also found that the networks trained with our approach become more robust. The intuition behind our proposed method is based on the observation that if the sample size of a class is too small, the network may tend to focus on the background instead of the object itself [41,42]. This can lead to undesirable biases in the classifier. In our approach, the constraint forces the network to not focus so much on the background.

The main contributions of this paper are:

- **Size Priors:** We propose Top-GAP, a regularization technique incorporating a size prior directly into the network architecture. This method constrains the number of pixels the network utilizes during training and inference. It is beneficial for object classification tasks in contexts without perspective projections, such as biomedical imaging and datasets with centered objects.
- **Effective Receptive Field (ERF):** We link Top-GAP to the ERF and measure the influence of the feature output on the background and object pixels. We show that Top-GAP directs the network's focus towards object pixels, reducing background influence.

- **Robustness to Adversarial Attacks and Distribution Shifts:** Further evidence that the background is less important is given by our robustness experiments. Top-GAP improves robustness against PGD and Square Attack, achieving up to 50% higher accuracy without adversarial training. It also enhances accuracy by up to 5% for datasets such as Waterbirds [42].
- **Intersection over Union:** By adjusting the pixel constraint, our method enables the network to focus more precisely on specific objects, leading to up to a 25% improvement in Intersection over Union (IOU) compared to GradCAM and Recipro-CAM [7].

2 Related Work

Our work is related to different strands of research, each dealing with different aspects of improving the features and robustness of neural networks. This section outlines these research directions and introduces their relevance to our novel approach.

Adversarial robustness. It has been shown that neural networks are susceptible to small adversarial perturbations of the image [15]. For this reason, many methods have been developed to defend against such attacks. Some methods use additional synthetic data to improve robustness [16,51]. [51] makes use of diffusion models, while [16] uses an external dataset. Other methods have shown that architectural decisions can influence robustness [21, 36]. A disadvantage of all these approaches is that the clean accuracy and training speed are negatively affected [10, 38].

Bias mitigation and guided attention. A notable line of research concentrates on channeling the network's focus towards specific feature subsets. Of concern is the prevalence of biases within classifiers, arising due to training on imbalanced data that perpetuates stereotypes [5]. Biases may also stem from an insufficient number of samples [4, 6, 55], causing the network to emphasize incorrect features or leading to problematic associations. For instance, when the ground truth class is "boat", the network might focus on waves instead of the intended object.

[18, 53] introduce training strategies to use CAMs as labels and refine the classifier's attention toward specific regions. In contrast, [39] proposes transforming the input images to mitigate biases tied to protected attributes like gender. Moreover, [26] suggests a method to uncover latent biases within image datasets.

Weakly-supervised semantic segmentation (WSSS). [25] focuses on accurate object segmentation given class labels. The Puzzle-CAM paper [23] introduces a novel training approach, which divides the image into tiles, enabling the network to concentrate on various segments of the object, enhancing segmentation performance. There are many more publications that focus on improving WSSS [45]. Some making use of foundational models such as Segment Anything Model (SAM) [24] or using multi-modal models like CLIP [37].

Priors. Prior knowledge is an important aspect for improving neural network predictions. For example, YOLOv2 [40] calculated the average width and height

of bounding boxes on the dataset and forced the network to use these boxes as anchors. However, there are many other works that have tried to use some prior information to improve predictions [8, 20, 35, 48, 57]. In particular, [35] has proposed to add constraints during the training of the network. For example, they propose a background constraint to limit the number of non-object pixels. However, they only train the coarse output heat maps with convex-constrained optimization. The problem is that the use of constraints can make it harder to find the global optima. Therefore, it is harder to train the whole network.

Our approach. Much like bias mitigation strategies and attention-guided techniques, we direct the network's focus to specific areas. However, our approach does not require segmentation labels and only minimally changes the CNN architectures. The objective is to maintain comparable clean accuracy and the number of parameters, while significantly improving the interpretability of objects. In contrast to WSSS, we do not intend to segment entire objects, but instead continue to concentrate on the most discriminative features. Given that we modify the classification network itself, we also diverge from methods that solely attempt to enhance CAMs of pretrained models.

3 Method

In most cases of image classification, the majority of pixels are not important for the prediction. Usually, only a small object in the image determines the class. Our approach is geared towards these cases. In contrast, many modern CNNs implicitly operate under the assumption that every pixel in an image can be relevant for identifying the class. This perspective becomes evident when considering the global average pooling (GAP) layer [27] used in modern CNNs. The aim of the GAP layer is to eliminate the width and height dimensions of the last feature matrix, thereby making it possible to apply a linear decision layer. The GAP layer averages all locations within the last feature matrix without making a distinction between the positions or values. This means that a corner position is treated in the same way as a center position. We also note that each of the locations in the last feature matrix corresponds to multiple pixels in the input image. This is known as the receptive field. Now, we want to define more formally the terminology.

Definition 1 (Effective receptive field). *Let $X^{(p)}_{i^{(p)},j^{(p)}}$ be the feature matrix on the pth layer for $1 \leq p \leq n$ with coordinates $(i^{(p)}, j^{(p)})$. The input to the neural network is at $p = 1$ and the output feature map at $p = n$. Then the effective receptive field (ERF) of the output location $(i^{(n)}, j^{(n)})$ with respect to the input pixel $(i^{(1)}, j^{(1)})$ is given by $\frac{\partial X^{(n)}_{i^{(n)},j^{(n)}}}{\partial X^{(1)}_{i^{(1)},j^{(1)}}}$ [31].*

This definition assumes that each layer has only a single channel. For multiple output channels, we compute $\sum_{k=1}^{c^{(n)}} \frac{\partial X^{(n)}_{i^{(n)},j^{(n)},k}}{\partial X^{(1)}_{i^{(1)},j^{(1)}}}$ where $c^{(n)}$ are the channels of

the last feature map. The ERF characterizes the impact of some input pixel on the output.

Definition 2 (Global Average Pooling). *The feature output of the neural network $X^{(n)}$ is averaged to obtain a single value. This operation is known as Global Average Pooling (GAP) and is defined as:*

$$GAP(X^{(n)}) = \frac{1}{h^{(n)}w^{(n)}} \sum_{i=1}^{h^{(n)}} \sum_{j=1}^{w^{(n)}} X_{i,j}^{(n)} ,$$

where $h^{(n)}$ is the height and $w^{(n)}$ is the width of the output feature map. In practice, there is not only one channel but $c^{(n)}$ channels.

An example shall explain the two terms. In case of EfficientNet-B0 [46], $X^{(n)}$ has dimension $7 \times 7 \times 1280$ for an input image of size 224×224 where $c^{(n)} = 1280$ are the channels. The GAP$(\cdot)$ operation reduces $X^{(n)}$ to a vector of size 1280×1. All of the 7×7 locations have an effect on the classification. With the help of the ERF, we can measure how much the 224^2 input pixels contribute to the 7^2 output locations.

Another method to analyze what the neural network focuses on are the so-called class activation maps. These methods modify $X^{(n)}$ so that we get a visualization of what is important for the neural network.

Definition 3 (Class Activation Map). *The product of multiplying the output tensor $X^{(n)}$ by some weight coefficient W is known as a class activation map (CAM) [56]. The standard CAM, also known as "CAM", uses the weights of the linear decision layer L.*

In the previous example, the linear decision layer L would map the 1280 channels to $c^{(n+1)}$ class channels. The output of the CAM would be in this case $7 \times 7 \times c^{(n+1)}$. Each of the $c^{(n+1)}$ maps can be upsampled to obtain a visualization.

Definition 4 (GradCAM). *GradCAM is a generalization of CAM to non-fully convolutional neural networks (non-FCN) such as VGG. It is equivalent to the standard CAM for FCN like ResNet. It is defined as follows*

$$GradCAM(X, c) = ReLU\left(\sum_k W_{k,c}, X_k\right) ,$$

with $W_{k,c} = GAP\left(\frac{\partial L(X)_c}{\partial X_k}\right)$, k being the channel index of X and c being the index of the linear layer. Usually the last feature map $X^{(n)}$ is chosen for X.

In addition to GradCAM, there are many other CAM methods. However, they are all based on reducing the channels of $X^{(n)}$ in order to obtain a visualization. Instead of improving GradCAM, as so many approaches have done before [9, 13, 22, 33, 49], we propose that the output of the CNN should be both

a CAM and a prediction. Then we can regularize the CAM during training and can more fundamentally influence what is highlighted in the CAM.

Our approach involves integrating an object size constraint directly into the network, designed to enforce the utilization of a limited set of pixels for classification. This constraint allows for noise reduction and the elimination of unnecessary pixels from the CAM. In cases where specific-sized features determine the class, we can incorporate this prior knowledge into the neural network, enhancing its classification accuracy.

Before introducing the object constraint, we first change the model structure to output a higher-resolution CAM.

3.1 Changing the model output structure

Figure 2 shows the general structure of our architecture. The backbone can be any standard CNN such as VGG [44], ResNet [17], ConvNeXt [30] or EfficientNet [46]. Depending on the backbone, we use the last 3 or 4 feature maps as input to a feature pyramid network (FPN) [28]. We note that the original FPN as used for object detection was simplified in order to reduce parameters. All the feature maps are upsampled to the size of the largest feature map and added together. We found no advantage in using concatenation. This output is given to a final convolutional layer that has the number of output classes as filters. Note that a convolutional layer with kernel size 1 is used for the implementation of the final linear layer. Optionally, dropout can be applied as regularization during training. Lastly, we employ Top-GAP to obtain a single probability for each class. Top-GAP is introduced in the following section.

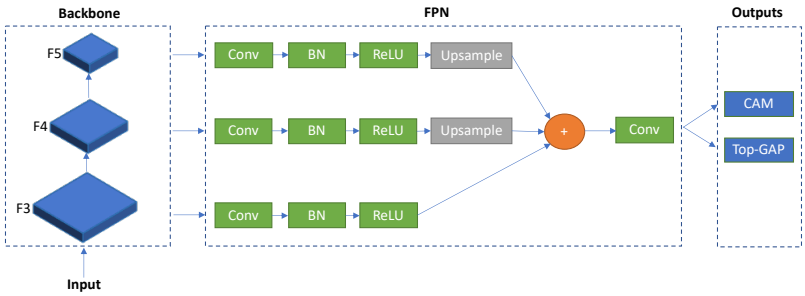


Fig. 2: Example of our architecture applied to a backbone with 3 feature maps (e.g. 7×7, 14×14, 28×28). For all convolutions except the final one, a kernel size of 3 and 256 filters is used. The last convolution employs a kernel size of 1, with the number of filters set to match the number of output classes. The CAM is as large as the biggest feature map (here F3). Our pooling layer ("Top-GAP") averages the CAMs given by the last convolutional layer ("Conv") to create a vector containing the probability for each class. For the CAM, we disable "Top-GAP" and perform min-max scaling.

For convenience, we explicitly define two modes for our model (refer to Fig. 2):

1. CAM: The output feature map is upsampled to the size of the input image and normalized to be in the range $[0, 1]$.
2. training/prediction: Top-GAP is enabled to obtain the probabilities for each class.

Without our modified model, we would need to use a method such as GradCAM to obtain a visualization.

Let us compare the two approaches: EfficientNet-B0 with GradCAM and EfficientNet-B0 with our output structure (see Fig. 2). GradCAM does not require any additional parameters because it generates the activation map from the model itself. If we change the model structure, we have more parameters, but also more influence on what is seen in the CAM. If we were to replace GradCAM with LayerCAM or some other method, it would never have the same impact as changing the model training itself (our approach). In addition, GradCAM does not combine multiple feature maps by default to achieve better localization.

In our approach, the standard output linear layer of some classification model like EfficientNet-B0 is substituted with $f + 1$ convolutional layers, where f corresponds to the number of feature maps. This leads to a small increase in the number of parameters.

Architecture	Params (unmodified)	Params (ours)
VGG11-BN	132.87M	12.43M
EfficientNet-B0	4.08M	4.75M
DenseNet-121	7.98M	8.03M

Table 1: Number of parameters for some architectures. We have fewer parameters than VGG because all additional linear layers are removed.

As indicated in Tab. 1, we can achieve a comparable number of parameters.

These changes to the model are prerequisites for enabling the integration of size constraints within the neural network. If only the last feature map were used, a single value would correspond to an excessively large area in the original image. Hence, combining multiple feature maps proves advantageous. This idea is reinforced by findings from [22], which highlight that employing multiple layers enhances the localization capabilities of CAMs.

3.2 Defining the pixel constraint (Top-GAP)

Instead of using the standard GAP layer, we replace the average pooling by a top-k pooling, where only the k highest values of the feature matrix are considered for averaging. This pooling layer limits the number of input pixels that the network can use for generating predictions.

In a standard CNN, the last feature map is at layer n. In our model (Fig. 2), the last feature map is at $n + 1$ because we replaced the linear decision layer L by a 1×1 convolution.

Definition 5 (Top-GAP). *We define the Top-GAP layer as follows:*

$$\textit{Top-GAP}(\tilde{X}, k)_t = \frac{1}{k} \sum_{i=1}^{k} \tilde{X}_{i,t} \,,$$

where $\tilde{X}$ represents the ordered feature matrix $X^{(n+1)}$ with dimensions $h^{(n+1)}w^{(n+1)} \times c^{(n+1)}$, where $c^{(n+1)}$ corresponds to the number of output classes. Each of the $c^{(n+1)}$ column vectors is arranged in descending order by value, and k values are selected. i indicates the ranking, with $i = 1$ being the largest value and $i = k$ being the smallest. t is an index indicating the channel. We select for each channel different values.

When $k = 1$, we obtain global max pooling (GMP). When $k = h^{(n+1)}w^{(n+1)}$, the layer returns to standard GAP. The parameter k enforces the pixel constraint, and its value depends on the image size. For instance, if the largest feature map has dimensions 56×56, then $\frac{k}{56^2}$ values are selected. Hence, when adjusting this parameter, it is crucial to consider the relative object size in the highest feature map.

3.3 Classification loss function

The last component of our method involves changing the loss function. While the Top-GAP$(\cdot)$ layer considers only locations with the highest values, these locations might not necessarily be the most important ones. Thus, it becomes essential to incentivize the reduction of less important positions to zero.

To achieve this, we add an ℓ_1 regularization term to the loss function, inducing sparsity in the output. The updated loss function is defined as follows:

$$L = \lambda ||X^{(n+1)}||_1 + \mathrm{CE}\left(\hat{y}, y\right) \,, \tag{1}$$

where $\mathrm{CE}(\hat{y}, y)$ represents the cross-entropy loss between the prediction $\hat{y} = \mathrm{softmax}\left(\text{Top-GAP}(\tilde{X}, k)\right)$ and the ground truth y. $\tilde{X}$ is the ordered $X^{(n+1)}$ feature output in our model, while k is a fixed non-trainable parameter. Here, λ controls the strength of the regularization. We found that for most datasets $\lambda = 1$ is sufficient.

4 Evaluation

In this section, we will systematically test the claims of our method on several datasets. Since there is no ground truth for explainability, we focus mainly on surrogate measures. Our main surrogate measure for interpretability is the background of the image. We show that our method causes the network to focus less on it. Furthermore, we also evaluate how our model behaves in the presence of distribution shifts. A description of the datasets used here can be found in the appendix A.

Our method consists of three components: Top-GAP, model structure, and loss function. Top-GAP has a hyperparameter k that defines the number of input pixels the network should use. We tested values $k \in \{64, 128, \ldots, 1024, 2048\}$ and only report the result that maximizes the metric (e.g. accuracy).

4.1 Hypothesis: the gradient of object pixels becomes more important

We want to show that with our method not all pixels in the input image have the same influence on the output feature map $X^{(n+1)}$. Recall that in our model, the last feature map is at $n+1$ because we have replaced the linear layer with a convolutional layer.

In most datasets (e.g. ImageNet), the object to be classified is located in the center of the input image. While each pixel in the input image corresponds to multiple values in the output feature map $X^{(n+1)}$, the general position is the same. The center in the output is also the center in the input.

The input pixels should contribute much more to the center than to the background of $X^{(n+1)}$. We want to quantify how much influence the input pixels have on the center of $X^{(n+1)}$ and on the corner of $X^{(n+1)}$. For this, we use Definition 1 and define a metric.

Definition 6 (ERF distance). *Let* $ERF(1,1) = \frac{1}{hw}\sum_{i,j}\left|\frac{\partial X^{(n+1)}_{1,1}}{\partial X^{(1)}_{i,j}}\right|$ *to be the absolute change of the output corner position* $(1,1)$ *with all input pixels* (i,j)*. Similarly, we define* $ERF(\frac{h}{2},\frac{w}{2})$ *to be the change of the output center position with respect to the input, where* h *and* w *is the width of the output feature map. Then the ERF distance is* $ERF(\frac{h}{2},\frac{w}{2}) - ERF(1,1)$*.*

Intuitively, we expect a low value for $\mathrm{ERF}(1,1)$ because the corner position of the feature map contains less information. Similarly, $\mathrm{ERF}(\frac{h}{2},\frac{w}{2})$ should be a high value because the object is in the center. If the difference between the two values is low, it means that each pixel contributes similarly to the output.

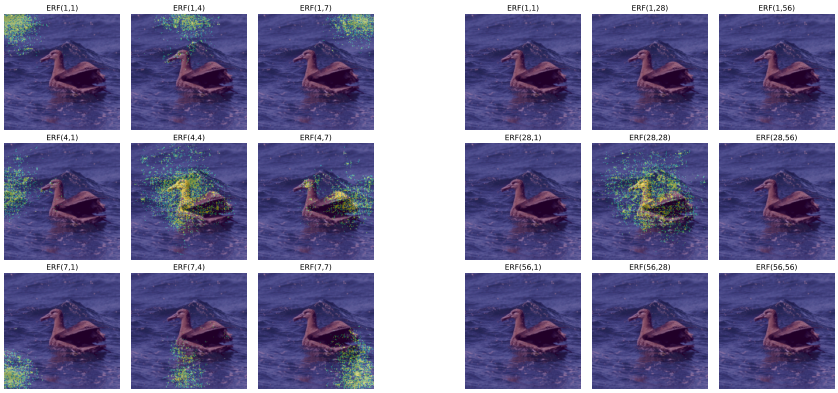
Table 2 shows that for the standard CNN the center of the image has the same effect as the corner. $\mathrm{ERF}(1,1)$ has the same value range as $\mathrm{ERF}(\frac{h}{2},\frac{w}{2})$. Compare this to our approach, where there is a large difference between the center and the corner ERF.

During backpropagation, the neural network goes from the end to the beginning of the network and updates the weights. With our method, we set the gradient at the background positions of the last feature matrix to almost zero. This also affects all other layers as a consequence of the chain rule.

The visualization in Fig. 3 confirms the numerical results. Since there are $7^2 = 49$ positions for the standard ResNet and 56^2, we only considered 9 pixel positions. We see that the gradient disappears at the locations where there is no object. More numerical details are provided in the appendix in Tab. A1 and Tab. A2.

Dataset	Arch	ERF distance ↑	ERF distance (ours) ↑
COCO [29]	EN	0.108	**0.447**
	CN	0.072	**0.288**
	RN	0.273	**0.399**
Oxford [34]	EN	0.013	**0.383**
	RN	0.060	**0.443**
CUB-200-2011 [47]	EN	-0.033	**0.480**
	CN	-0.034	**0.242**
	RN	0.092	**0.529**

Table 2: The table shows that our approach leads to a different ERF. The center has a stronger effect than the corner of the image. "Ours" is our approach (with pixel constraint, ℓ_1 loss and the changes to the model). The other column is the standard model without any changes. EN = EfficientNet-B0, CN = ConvNeXt-tiny, RN = ResNet-18.



(a) Standard ResNet-18 **(b)** ResNet-18 with our approach

Fig. 3: ERF for various locations in the output feature map. The background becomes less important using our approach. The last feature map of standard ResNet has size 7×7, with our approach it has size 56×56.

4.2 Hypothesis: the background is less susceptible to adversarial attacks

The last experiment showed that by changing the values in the output feature matrix, we also change the effect of the input pixels. Another idea to prove that the network focuses less on the background is to use adversarial attacks. The goal of such an attack is to change the input pixels so that the classification prediction is different. Using our method, we expect the attack area to be smaller since the network uses the background less.

It is important to emphasize that we only need to achieve higher robustness against standard models. Our goal here is to force the network to focus on different regions in order to achieve better interpretability. We do not intend to

compete with adversarially trained networks. Adversarial training (AT) is slower, leads to less clean accuracy and does not make the networks more interpretable.

Method	Arch	PGD^{20} ↑	PGD^{50} ↑	Square ↑	Clean ↑
Standard	PRN18	0.0	0.0	0.0	0.945
Top-GAP (ours)	PRN18	0.517	0.313	0.343	**0.951**
FGSM-AT [2]	PRN18	-	**0.476**	-	0.81
SAT [36]	RN50	**0.552**	-	-	0.849

Table 3: Results on CIFAR-10. We use $\ell_\infty = 8/255$ and 20/50 steps (PGD). For AT models, we report the values from the papers. SAT = Standard Adversarial Training, PRN18 = PreAct ResNet-18, RN50 = ResNet-50. Our results are close to the robustness of adversarially trained networks.

Tab. 3 shows that we outperform the standard models by far and even achieve comparable clean accuracy. Notably, square attack [1] does not rely on local gradient information. It should, therefore, be not affected by gradient masking. This shows that our robustness is not necessarily a result of "shattered gradients" [3]. On other datasets, such as ImageNet, we similarly see small increases in robustness while keeping the same accuracy (refer to the appendix Tab. A3).

To make the argument that the network is less susceptible to attacks to the background even more convincing, we consider the FGSM attack [14]. It uniformly perturbs each pixel by $\pm k$ for some $1 \leq k \leq 255$. Instead of perturbing the whole image, we perturb either just the object or just the background. For this, we use the segmentation mask provided by the CUB dataset.

Definition 7 (Attack distance). *Let $SAR(I) = 1 - \text{Acc}$ be the successful attack rate, given perturbed images I. We define $SAR(O) - SAR(B)$ to be the attack distance (AD) between the object image O and the background image B.*

In image O, only the pixels of the object were changed by ± 1, while all other pixels of the original image were retained. Similarly, in image B, only the background was perturbed while the object remained untouched. Just like the ERF distance, we expect $\text{SAR}(O)$ to be large and $\text{SAR}(B)$ to be small. FGSM should be more successful if it attacks the object and less successful if it attacks the background.

Arch	Method	Attack distance
EN	Standard	0.016
	ours	**0.064**
RN	Standard	0.022
	ours	**0.065**
CN	Standard	0.078
	ours	**0.132**

Table 4: The background is less susceptible to attacks with our approach. The dataset is CUB-200-2011.

In Tab. 4, we see that when using the standard networks, the values of $\text{SAR}(O)$ and $\text{SAR}(B)$ are close to each other. This means that the center has the same effect as the background. The network concentrates on all pixels equally. With our approach, we can manipulate the class more easily by changing the object pixels.

Background pixels, on the other hand, are less important. For this reason, we have a higher AD value. This gives a second proof for our hypothesis.

4.3 Hypothesis: classification makes use of object pixels

Another way to show that we are directing the attention of the network to the object is through distribution shifts. To address this, we use the Waterbirds dataset [42], where the backgrounds of images are replaced. Furthermore, we evaluate accuracy on ImageNet-Sketch [50] and ImageNet-C [19].

Dataset	Arch	Acc ↑	Acc ↑ (ours)
CUB → Waterbirds	EN	0.521	**0.564**
	CN	0.722	**0.737**
	RN	0.468	**0.520**
ImageNet → Sketch	VG	0.179	**0.200**
	RN	0.206	**0.236**
ImageNet → ImageNet-C	VG	0.494	**0.498**
	RN	0.513	**0.535**

Table 5: Evaluation of the out-of-distribution accuracy by using images outside the original dataset. $X \rightarrow Y$ means train on X and validate on Y.

An improvement in accuracy can be observed for all datasets. While there are works that show higher accuracy for datasets such as ImageNet-Sketch [12], they are based on specialized training methods (self-supervised, semi-supervised) and/or more data. Our proposed method comes "without cost" in the sense that it works for any architecture and dataset, without requiring more GPU resources. It can be viewed as a regularization technique. This increase in robustness does not negatively affect the accuracy. We also see a comparable accuracy when using 5-fold stratified cross validation (refer to Tab. 6).

4.4 Hypothesis: increased interpretability due to pixel constraints

The last experiments have shown that we can direct the attention of the network to the object. So far, we have used the pixel constraint k, which maximizes the metric. However, it is also possible to vary this value to incorporate human knowledge into the prediction. In Fig. 4, we measure the sparsity of our CAM.

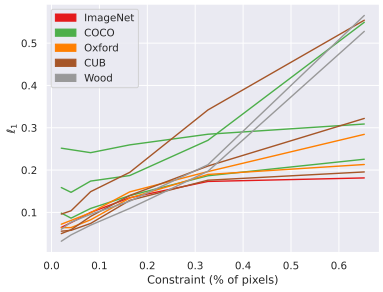


Fig. 4: Each line in the graph represents a dataset+architecture combination. The x-axis shows the normalized k value (e.g. $\frac{64}{56^2}$) for the constraint, while the y-axis represents the ℓ_1 norm.

Dataset	Arch	Accuracy ↑	Accuracy (ours) ↑
COCO	EN	0.801 ± 0.009	**0.803** ± 0.006
	CN	0.939 ± 0.006	**0.940** ± 0.005
	RN	0.853 ± 0.004	**0.868** ± 0.005
Wood	EN	0.672 ± 0.037	**0.681** ± 0.041
	CN	0.721 ± 0.030	**0.724** ± 0.033
Oxford	EN	0.854 ± 0.008	**0.863** ± 0.010
	RN	0.861 ± 0.007	**0.862** ± 0.007
CUB	EN	0.76 ± 0.01	**0.77** ± 0.005
	RN	**0.69** ± 0.014	0.685 ± 0.006
	CN	**0.862** ± 0.007	0.854 ± 0.005
ImageNet	VG	**0.704**	0.699
	RN	**0.698**	0.697

Table 6: Our approach refers to the changed model with pixel constraint and ℓ_1 loss. The original models come from PyTorch Image Models [52] and are pretrained on ImageNet. EN = EfficientNet-B0, CN = ConvNeXt-tiny, RN = ResNet-18, VG = VGG11-bn. For ImageNet, we only use a train/val split.

It is evident that as we increase the constraint k, the number of displayed pixels in the CAM also rises. More experiments are provided in Tab. A4 and Appendix E in the appendix.

We evaluated our approach on a real-world dataset with microscopic images. Fig. 5 shows that we can use our constraint to direct the focus of the network to the vessels. The standard model focuses on the background or fibers instead. For biologists, however, only the vessels are important.

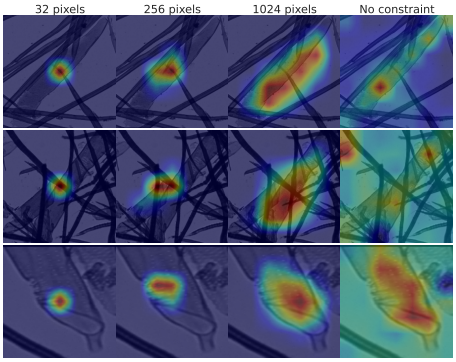


Fig. 5: Impact of pixel constraint on CAM (Wood identification dataset [32]). "No constraint" denotes a standard unmodified EfficientNet-B0 model using CAM/GradCAM [43]. The object in the center, known as a vessel should be highlighted. Without our method, the background containing fibers is also highlighted.

Finally, we assess the segmentation overlap. While segmentation masks should not be considered as ground truth for explainability, they provide valuable insights into the focus of the network. This analysis allows us to quantitatively measure whether the network is predominantly focused on the object of interest or on the surrounding background.

Dataset	Arch	IOU (GC) ↑	IOU (RC) ↑	IOU (ours) ↑
COCO	EN	0.309	0.245	**0.348**
	CN	0.103	0.268	**0.361**
	RN	0.371	0.359	**0.391**
CUB	EN	0.323	0.337	**0.414**
	CN	0.125	0.279	**0.389**
	RN	0.268	0.34	**0.435**

Table 7: Comparison of Intersection over Union (IOU) scores across different methods and architectures. The IOU (GC) column represents the standard unchanged model using GradCAM (GC). Similarly, RC is Recipro-CAM [7].

Tab. 7 gives even more evidence that the pixel constraint allows the network to focus more on the object.

5 Discussion and Outlook

In this paper, we presented a new approach to improve the explainability of CNNs. Our method focuses on controlling the number of pixels a network can use for predictions, resulting in CAMs with lower noise and better localization. The results show that our approach is effective on a variety of datasets and architectures. We have consistently observed both visually and numerically more concise feature representations in the CAMs. In addition, our approach provides a novel form of network regularization. By forcing the network to focus exclusively on objects of a predefined size, we reduce the risk of highlighting irrelevant regions, which can be critical for applications that require precise object localization or for reducing bias.

Limitations. Determining the optimal value for the pixel constraint parameter k currently depends on hyperparameter tuning. It is possible to explore automated methods for determining this parameter to improve efficiency and adaptability. Second, given the variety of object sizes, it may not be ideal to rely on a single parameter for all objects. Only in specific areas such as biomedical imaging, where object size are not influenced by perspective projections (e.g. microscope) typically show low size variances. Investigating ways to dynamically adjust this parameter for different object sizes would be a valuable line of research. Finally, the proposed FPN module can be further refined to improve accuracy even more.

References

1. Andriushchenko, M., Croce, F., Flammarion, N., Hein, M.: Square attack: A query-efficient black-box adversarial attack via random search. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J. (eds.) Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXIII. Lecture Notes in Computer Science, vol. 12368, pp. 484–501. Springer (2020). https://doi.org/10.1007/978-3-030-58592-1_29, https://doi.org/10.1007/978-3-030-58592-1_29
2. Andriushchenko, M., Flammarion, N.: Understanding and improving fast adversarial training (2020)
3. Athalye, A., Carlini, N., Wagner, D.: Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples (2018)
4. Bolukbasi, T., Chang, K.W., Zou, J., Saligrama, V., Kalai, A.: Man is to computer programmer as woman is to homemaker? debiasing word embeddings (2016)
5. Buolamwini, J., Gebru, T.: Gender shades: Intersectional accuracy disparities in commercial gender classification. In: Friedler, S.A., Wilson, C. (eds.) Proceedings of the 1st Conference on Fairness, Accountability and Transparency. Proceedings of Machine Learning Research, vol. 81, pp. 77–91. PMLR (23–24 Feb 2018), https://proceedings.mlr.press/v81/buolamwini18a.html
6. Burns, K., Hendricks, L.A., Saenko, K., Darrell, T., Rohrbach, A.: Women also snowboard: Overcoming bias in captioning models (2019)
7. Byun, S.Y., Lee, W.: Recipro-cam: Fast gradient-free visual explanations for convolutional neural networks (2023)
8. Cai, J., Hou, J., Lu, Y., Chen, H., Kneip, L., Schwertfeger, S.: Improving cnn-based planar object detection with geometric prior knowledge. In: 2020 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR). pp. 387–393 (2020). https://doi.org/10.1109/SSRR50563.2020.9292601
9. Chattopadhay, A., Sarkar, A., Howlader, P., Balasubramanian, V.N.: Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks. In: 2018 IEEE Winter Conference on Applications of Computer Vision (WACV). IEEE (mar 2018). https://doi.org/10.1109/wacv.2018.00097
10. Clarysse, J., Hörrmann, J., Yang, F.: Why adversarial training can hurt robust accuracy (2022)
11. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 248–255 (2009). https://doi.org/10.1109/CVPR.2009.5206848
12. Fang, Y., Wang, W., Xie, B., Sun, Q., Wu, L., Wang, X., Huang, T., Wang, X., Cao, Y.: Eva: Exploring the limits of masked visual representation learning at scale (2022)
13. Fu, R., Hu, Q., Dong, X., Guo, Y., Gao, Y., Li, B.: Axiom-based grad-cam: Towards accurate visualization and explanation of cnns. CoRR **abs/2008.02312** (2020), https://arxiv.org/abs/2008.02312
14. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples (2014). https://doi.org/10.48550/ARXIV.1412.6572, https://arxiv.org/abs/1412.6572
15. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples (2015)

16. Gowal, S., Rebuffi, S., Wiles, O., Stimberg, F., Calian, D.A., Mann, T.A.: Improving robustness using generated data. CoRR **abs/2110.09468** (2021), https://arxiv.org/abs/2110.09468
17. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition (2015)
18. He, Y., Yang, X., Chang, C.M., Xie, H., Igarashi, T.: Efficient human-in-the-loop system for guiding dnns attention (2023)
19. Hendrycks, D., Dietterich, T.G.: Benchmarking neural network robustness to common corruptions and perturbations. CoRR **abs/1903.12261** (2019), http://arxiv.org/abs/1903.12261
20. Hou, W., Tao, X., Xu, D.: Combining prior knowledge with cnn for weak scratch inspection of optical components. IEEE Transactions on Instrumentation and Measurement **70**, 1–11 (2021). https://doi.org/10.1109/TIM.2020.3011299
21. Huang, H., Wang, Y., Erfani, S.M., Gu, Q., Bailey, J., Ma, X.: Exploring architectural ingredients of adversarially robust deep neural networks (2022)
22. Jiang, P.T., Zhang, C.B., Hou, Q., Cheng, M.M., Wei, Y.: Layercam: Exploring hierarchical class activation maps for localization. IEEE Transactions on Image Processing **30**, 5875–5888 (2021). https://doi.org/10.1109/TIP.2021.3089943
23. Jo, S., Yu, I.J.: Puzzle-CAM: Improved localization via matching partial and full features. In: 2021 IEEE International Conference on Image Processing (ICIP). IEEE (sep 2021). https://doi.org/10.1109/icip42928.2021.9506058, https://doi.org/10.1109/icip42928.2021.9506058
24. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., Dollár, P., Girshick, R.: Segment anything (2023)
25. Li, K., Wu, Z., Peng, K., Ernst, J., Fu, Y.: Tell me where to look: Guided attention inference network. CoRR **abs/1802.10171** (2018), http://arxiv.org/abs/1802.10171
26. Li, Z., Xu, C.: Discover the unknown biased attribute of an image classifier. CoRR **abs/2104.14556** (2021), https://arxiv.org/abs/2104.14556
27. Lin, M., Chen, Q., Yan, S.: Network in network (2014)
28. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection (2017)
29. Lin, T.Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C.L., Dollár, P.: Microsoft coco: Common objects in context (2015)
30. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A convnet for the 2020s (2022)
31. Luo, W., Li, Y., Urtasun, R., Zemel, R.: Understanding the effective receptive field in deep convolutional neural networks (2017)
32. Nieradzik, L., Sieburg-Rockel, J., Helmling, S., Keuper, J., Weibel, T., Olbrich, A., Stephani, H.: Automating wood species detection and classification in microscopic images of fibrous materials with deep learning (2023)
33. Omeiza, D., Speakman, S., Cintas, C., Weldemariam, K.: Smooth grad-cam++: An enhanced inference level visualization technique for deep convolutional neural network models. CoRR **abs/1908.01224** (2019), http://arxiv.org/abs/1908.01224
34. Parkhi, O.M., Vedaldi, A., Zisserman, A., Jawahar, C.V.: Cats and dogs. In: IEEE Conference on Computer Vision and Pattern Recognition (2012)
35. Pathak, D., Krähenbühl, P., Darrell, T.: Constrained convolutional neural networks for weakly supervised segmentation (2015)

36. Peng, S., Xu, W., Cornelius, C., Hull, M., Li, K., Duggal, R., Phute, M., Martin, J., Chau, D.H.: Robust principles: Architectural design principles for adversarially robust cnns (2023)
37. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision (2021)
38. Raghunathan, A., Xie, S.M., Yang, F., Duchi, J.C., Liang, P.: Adversarial training can hurt generalization. CoRR **abs/1906.06032** (2019), http://arxiv.org/abs/1906.06032
39. Rajabi, A., Yazdani-Jahromi, M., Garibay, O.O., Sukthankar, G.: Through a fair looking-glass: mitigating bias in image datasets (2022)
40. Redmon, J., Farhadi, A.: Yolo9000: Better, faster, stronger (2016)
41. Ribeiro, M.T., Singh, S., Guestrin, C.: "why should i trust you?": Explaining the predictions of any classifier (2016)
42. Sagawa, S., Koh, P.W., Hashimoto, T.B., Liang, P.: Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. CoRR **abs/1911.08731** (2019), http://arxiv.org/abs/1911.08731
43. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-CAM: Visual explanations from deep networks via gradient-based localization. International Journal of Computer Vision **128**(2), 336–359 (oct 2019). https://doi.org/10.1007/s11263-019-01228-7
44. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2015)
45. Sun, W., Liu, Z., Zhang, Y., Zhong, Y., Barnes, N.: An alternative to wsss? an empirical study of the segment anything model (sam) on weakly-supervised semantic segmentation problems (2023)
46. Tan, M., Le, Q.V.: Efficientnet: Rethinking model scaling for convolutional neural networks (2020)
47. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The Caltech-UCSD Birds-200-2011 Dataset (Jul 2011)
48. Wang, C., Siddiqi, K.: Differential geometry boosts convolutional neural networks for object detection. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 1006–1013 (2016). https://doi.org/10.1109/CVPRW.2016.130
49. Wang, H., Wang, Z., Du, M., Yang, F., Zhang, Z., Ding, S., Mardziel, P., Hu, X.: Score-cam: Score-weighted visual explanations for convolutional neural networks (2020)
50. Wang, H., Ge, S., Xing, E.P., Lipton, Z.C.: Learning robust global representations by penalizing local predictive power. CoRR **abs/1905.13549** (2019), http://arxiv.org/abs/1905.13549
51. Wang, Z., Pang, T., Du, C., Lin, M., Liu, W., Yan, S.: Better diffusion models further improve adversarial training (2023)
52. Wightman, R.: Pytorch image models. https://github.com/rwightman/pytorch-image-models (2019). https://doi.org/10.5281/zenodo.4414861
53. Yang, X., Wu, B., Sato, I., Igarashi, T.: Directing dnns attention for facial attribution classification using gradient-weighted class activation mapping. CoRR **abs/1905.00593** (2019), http://arxiv.org/abs/1905.00593
54. Zarándy, Á., Rekeczky, C., Szolgay, P., Chua, L.O.: Overview of cnn research: 25 years history and the current trends. In: 2015 IEEE International Symposium on Circuits and Systems (ISCAS). pp. 401–404. IEEE (2015)

55. Zhao, J., Wang, T., Yatskar, M., Ordonez, V., Chang, K.W.: Men also like shopping: Reducing gender bias amplification using corpus-level constraints. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. pp. 2979–2989. Association for Computational Linguistics, Copenhagen, Denmark (Sep 2017). https://doi.org/10.18653/v1/D17-1323, https://aclanthology.org/D17-1323
56. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Learning deep features for discriminative localization (2015)
57. Zhou, X., Zhu, M., Pavlakos, G., Leonardos, S., Derpanis, K.G., Daniilidis, K.: Monocap: Monocular human motion capture using a cnn coupled with a geometric prior. IEEE Transactions on Pattern Analysis and Machine Intelligence **41**(04), 901–914 (apr 2019). https://doi.org/10.1109/TPAMI.2018.2816031

Top-GAP: Integrating Size Priors in CNNs for more Interpretability, Robustness, and Bias Mitigation

Supplementary Material

The appendix contains the following additional materials:

- A detailed description of the datasets: A.
- More details regarding the effective receptive field: Tab. A1, Tab. A2.
- More details regarding adversarial attacks: Tab. A3.
- More details regarding sparsity: Tab. A4, Tab. A5, Tab. A6, Tab. A7.

A Description of datasets

We test all our models on the following datasets:

- COCO [29]: We turned this segmentation dataset into a classification dataset by excluding images with more than one object. Furthermore, we kept only classes with a minimum of 20 samples per class. The resulting subset comprises 53 classes.
- Wood identification dataset [32]: This dataset consists of high-resolution microscopy images for hardwood fiber material. Nine distinct wood species have to be distinguished.
- Oxford-IIIT Pet Dataset [34]: The task is to differentiate among 37 breeds of dogs and cats.
- CUB-200-2011 [47] and Waterbirds [42]: 200 classes of birds have to be distinguished. Waterbirds replaces the background of the original images to test the models for biases.
- ImageNet [11]: A large-scale dataset with 1000 different classes. ImageNet-Sketch [50] / ImageNet-C [19] replaces the original validation images with out-of-distribution / corrupted images.
- CIFAR10: A dataset where each image has a size of 32×32. 10 classes have to be distinguished.

B Effective Receptive Field (ERF)

In Tab. 2, we only showed the difference between the center and the corner ERF. In the following tables, we provide the individual values. The gradients were z-normalized to have mean at 0 and standard deviation at 1.

Dataset	Architecture	Center ERF ↑	Center ERF (ours) ↑
COCO [29]	EfficientNet-B0	0.534	0.54
	ConvNeXt-tiny	0.47	0.439
	ResNet-18	0.595	0.571
Oxford [34]	EfficientNet-B0	0.087	0.51
	ResNet-18	0.104	0.542
CUB-200-2011 [47]	EfficientNet-B0	0.489	0.493
	ConvNeXt-tiny	0.477	0.398
	ResNet-18	0.538	0.534
Average	-	0.412	**0.503**

Table A1: Center ERF. "Ours" is our approach (with pixel constraint, ℓ_1 loss, and changes to the model). The other columns are the standard models without any changes.

The table shows that for the center pixel the gradient with respect to the input image is higher, when using our method.

Dataset	Architecture	Corner ERF ↓	Corner ERF (ours) ↓
COCO [29]	EfficientNet-B0	0.426	0.093
	ConvNeXt-tiny	0.398	0.151
	ResNet-18	0.322	0.172
Oxford [34]	EfficientNet-B0	0.074	0.127
	ResNet-18	0.044	0.099
CUB-200-2011 [47]	EfficientNet-B0	0.522	0.013
	ConvNeXt-tiny	0.511	0.156
	ResNet-18	0.446	0.005
Average	-	0.343	**0.102**

Table A2: Corner ERF. The values are lower using our approach, indicating improved performance. EN = EfficientNet-B0, CN = ConvNeXt-tiny, RN = ResNet-18.

Similarly, we see that the pixels have less of an effect when the corner of the input image is considered.

C Adversarial Attacks

We evaluated our approach using adversarial attacks on various datasets. For all the experiments summarized in Tab. A3, we use an ℓ_∞ constraint of $1/255$ for both FGSM and PGD attacks. The PGD attacks were performed with 40 steps. The $\pm$ symbol denotes the standard deviation of the accuracy across 5 different folds. Due to computational complexity, we report only a single run for the ImageNet dataset.

Dataset	Architecture	FGSM ↑	FGSM ↑ (ours)	PGD ↑	PGD ↑ (ours)	Clean Acc	Clean Acc (ours)
COCO	EfficientNet-B0	0.07	**0.3063**	0.0	**0.1098**	0.801 ± 0.009	**0.803** ± 0.006
	ConvNeXt-tiny	0.51	**0.678**	0.301	**0.463**	0.939 ± 0.006	**0.940** ± 0.005
	ResNet-18	0.288	**0.394**	0.08	**0.142**	0.853 ± 0.004	**0.868** ± 0.005
Wood	EfficientNet-B0	0.0	**0.277**	0.0	**0.085**	0.672 ± 0.037	**0.681** ± 0.041
	ConvNeXt-tiny	0.0	**0.404**	0.0	**0.01**	0.721 ± 0.030	**0.724** ± 0.033
Oxford	EfficientNet-B0	0.037	**0.107**	**0.0**	**0.0**	0.854 ± 0.008	**0.863** ± 0.010
	ResNet-18	0.104	**0.281**	0.016	**0.104**	0.861 ± 0.007	**0.862** ± 0.007
CUB	EfficientNet-B0	0.04	**0.147**	0.0	**0.04**	0.76 ± 0.01	**0.77** ± 0.005
	ResNet-18	0.06	**0.212**	0.0	**0.111**	**0.69** ± 0.014	0.685 ± 0.006
	ConvNeXt-tiny	0.134	**0.314**	0.03	**0.158**	**0.862** ± 0.007	0.854 ± 0.005
ImageNet	VGG11-bn	0.029	**0.217**	0.0	**0.01**	**0.704**	0.699
	ResNet-18	0.065	**0.256**	0.0	**0.059**	**0.698**	0.697

Table A3: Adversarial attack results comparing original and modified models with pixel constraint and ℓ_1 loss. The original models come from PyTorch Image Models [52] and pretrained on ImageNet.

While our approach does not achieve the robustness of adversarially trained networks, it demonstrates improved performance compared to standard networks.

D Sparsity

For almost all datasets and architectures, our approach achieved sparser CAMs. We see especially large decreases for ImageNet.

Dataset	Architecture	$\ell_1 \downarrow$	$\ell_1 \downarrow$ (ours)
COCO [29]	EfficientNet-B0	0.179	**0.064**
	ConvNeXt-tiny	0.251	**0.151**
	ResNet-18	**0.173**	0.194
Wood [32]	EfficientNet-B0	0.190	**0.032**
	ConvNeXt-tiny	0.110	**0.046**
Oxford [34]	EfficientNet-B0	0.154	**0.072**
	ResNet-18	0.151	**0.064**
CUB-200-2011 [47]	EfficientNet-B0	0.235	**0.05**
	ConvNeXt-tiny	0.164	**0.096**
	ResNet-18	0.121	**0.056**
ImageNet [11]	VGG11-BN	0.279	**0.064**
	ResNet-18	0.387	**0.123**

Table A4: The last column reports the sparsity of the CAM using our approach (with pixel constraint, ℓ_1 loss, and changes to the model). The third column is a standard model without any changes. For the standard model, we use GradCAM.

Only the lowest ℓ_1 of the different k values is reported. We observe that in general a strong pixel constraint such as $k = 64$ pixels leads to the lowest ℓ_1 value.

E Effect of ℓ_1 normalization on robustness and interpretability

ℓ_1 regularization has a strong influence on the results. Here we want to show that without the other components we would have lower interpretability, robustness and/or accuracy.

We consider the following variants of ResNet-18:

- ℓ_1 regularization only on the last feature output (activations) with $\lambda = 1.0$
- ℓ_1 regularization only on the last feature output (activations) with $\lambda = 0.1$
- ℓ_1 regularization on all activations with $\lambda = 10^{-3}$
- ℓ_1 regularization on all activations with $\lambda = 10^{-5}$
- ours: our approach

We use the CUB-200-2011 dataset. λ denotes the strength of the regularization.

E.1 Interpretability and accuracy

First, we analyze the effective receptive field and accuracy.

Approach	λ	Center ERF ↑	Corner ERF ↓	Accuracy
last	1.0	0.514	**0.002**	0.63
last	0.1	**0.536**	0.113	**0.69**
all	10^{-5}	0.488	0.416	**0.69**
all	10^{-3}	0.335	0.26	0.19
ours	1.0	**0.534**	**0.005**	**0.69**

Table A5: ℓ_1 regularization is a tradeoff between accuracy and ERF for the other approaches.

We see that we are only able to influence the ERF by regularizing the last feature map. While the approach "last + $\lambda = 1.0$" also achieves the same ERF as "ours", we see a significant decrease in accuracy of about 6%. Instead, we can also decrease λ, then the accuracy is the same, but we lose interpretability.

Additionally, without our Top-GAP pooling, we can no longer control the number of pixels. The λ parameter cannot be used for that.

Let X be the last feature output. We measure how many pixels are highlighted in the output, when adjusting λ and our Top-GAP k pixel constraint.

Approach	λ	$\|\|X\|\|_1$	Accuracy ↑
last	1.0	0.141	0.63
last	0.1	0.152	**0.69**
all	10^{-5}	0.119	**0.69**
all	10^{-3}	0.389	0.19
ours, $k = 128$	1.0	0.057	0.66
ours, $k = 256$	1.0	0.072	0.67
ours, $k = 512$	1.0	0.126	0.68
ours, $k = 1024$	1.0	0.174	**0.69**
ours, $k = 2048$	1.0	0.193	0.68

Table A6: As we increase the constraint value k, the number of pixels increases. The same behavior is not possible using λ. The accuracy would suffer too much.

When we increase the regularization strength from $\lambda = 0.1$ to $\lambda = 1.0$, the number of pixels only decreases from 0.152 to 0.141. However, the accuracy decreases by 6%.

Compare this to our approach. We can decrease the number of pixels while keeping the accuracy at the same level.

E.2 Robustness

Next, we analyze the level of robustness with respect to ℓ_1 regularization.

Approach	λ	PGD^{40} ↑	FGSM
last	1.0	0.06	0.15
last	0.1	0.03	0.11
all	10^{-5}	0.0	0.06
all	10^{-3}	0.0	0.03
ours	1.0	**0.11**	**0.21**

Table A7: Regularizing the last layer leads to the highest level of robustness. Our approach surpasses a simple regularization.

Regularizing only the last layer also brings a certain degree of robustness, but it comes at a price. The accuracy is lower and we still do not achieve the same level of sparsity for $\lambda = 1.0$ as with our approach.