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

Anonymous authors
Paper under double-blind review

Abstract

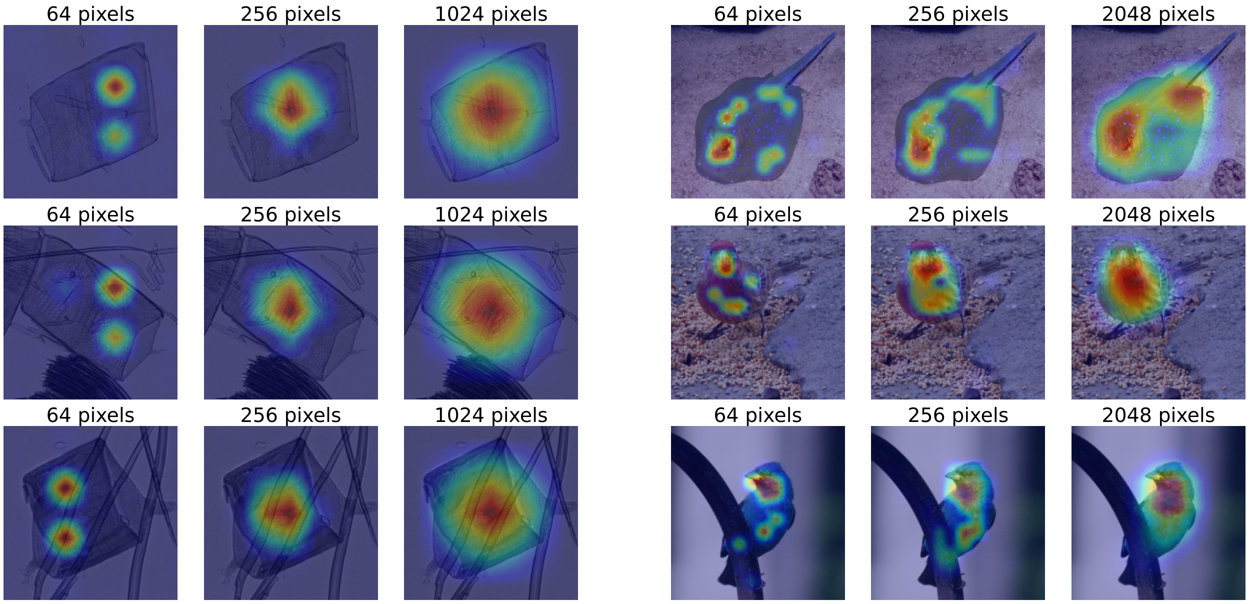
In the field of computer vision, convolutional neural networks (CNNs) have shown remarkable capabilities and are excelling in various tasks from image classification to semantic segmentation. However, their vulnerability to adversarial attacks remains a pressing issue that limits their use in safety-critical domains. In this paper, we present Top-GAP – a method that aims to increase the native robustness of CNNs by restricting the spatial size of feature representations. The advantage of our approach over common adversarial training is that our method does not degrade in clean accuracy or training speed. On CIFAR-10 with PGD $\epsilon = 8/255$ and 20 iterations, we achieve over 50% robust accuracy while retaining the original clean accuracy. Moreover, our size constraint helps to generate sparser and less noisy class activation maps, which significantly improves object localization and mitigates potential biases. We demonstrate on a variety of datasets and architectures that our method has comparable clean accuracy to regular trained models while improving localization and robustness. In addition, our method provides the ability to incorporate prior human knowledge about object sizes into the network, which is particularly beneficial in biological and medical domains where the variance in object sizes is not dominated by perspective projections.

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 (Zarándy et al., 2015). However, the robustness of these models remains a critical problem (DBL, 2018).

Many previous attempts to improve robustness have focused on adversarial training and additional (synthetic) images (Wang et al., 2023; Gowal et al., 2021). The disadvantage of these approaches is that both the computational complexity of the training drastically increases and the clean accuracy typically suffers (Clarysse et al., 2022; Raghunathan et al., 2019). A representative example is given by Peng et al. (2023), where standard adversarial training improves the robust accuracy on CIFAR-10 from 0% to 50.94% for ResNet-50, but the clean accuracy decreases from around 95% to 84.91%. Another example is Wang et al. (2023), where 50M training samples were generated, which inevitably leads to a strong increase in training time.

We propose a different approach that focuses on a novel method to regularize the network during training without adversarial samples. 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 Pathak et al. (2015), we do not need KKT conditions or the Lagrangian. The disadvantage of direct constrained optimization is that it can make gradient descent fail to 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.



(a) Wood identification

(b) ImageNet

Figure 1: Example images from a biological classification dataset (a) and ImageNet (b), where we limit the number of pixels (i.e. 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.

Restricting the output feature maps fundamentally changes the way the network works internally. In fig. 1, we see an example on how the constraint also affects the class activation map (CAM). We 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 (Sagawa et al., 2019; Ribeiro et al., 2016). 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:

- We introduce Top-GAP, a new approach to regularize networks by including a size prior in the network, which does not rely on the Lagrangian. This prior allows us to limit the number of pixels the network can use during inference. We show that this is especially beneficial for object classification tasks in imaging setups without perspective projections, such as biomedical imaging or benchmark datasets with centered objects.
- Extensive experiments on various architectures and datasets show, that our method improves robustness to adversarial attacks while maintaining high clean accuracy. For attacks such as FGSM/PGD, we achieve an increase in native robustness of over 50% accuracy without adversarial training.
- Further, our evaluation shows, that even in the case of imaging setups with strong object size variations, we can still find a size constraint leading to improvements over baseline settings.
- We also show that our method consistently improves the localization of objects. Depending on the dataset, we see an increase of up to 20% in IOU compared to GradCAM. This effect is even stronger when we measure the ℓ_1 of the CAMs.
- Finally, we report strong indications that our approach has the potential to mitigate bias. For example, when we take distribution shifts into account, we can achieve improvements in accuracy of up to 5%.

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 (Goodfellow et al., 2015). For this reason, many methods have been developed to defend against such attacks. Some methods use additional synthetic data to improve robustness (Wang et al., 2023; Gowal et al., 2021). Wang et al. (2023) makes use of diffusion models, while Gowal et al. (2021) uses an external dataset. Other methods have shown that architectural decisions can influence robustness (Peng et al., 2023; Huang et al., 2022). For example, the Transformer-style patchify stem is less robust than a classical convolutional stem. A disadvantage of all these approaches is that the clean accuracy and training speed are negatively affected (Raghunathan et al., 2019; Clarysse et al., 2022). "Native robustness" (Grabinski et al., 2022) on the other hand, is defined in literature as robustness which is achieved by architectural changes only. The problem with regular adversarial training is also that the networks are usually only robust against a single perturbation type Tramèr & Boneh (2019). Hence, methods not using adversarial training are less expensive and less prone to overfitting on specific attacks.

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 (Buolamwini & Gebru, 2018). Biases may also stem from an insufficient number of samples (Burns et al., 2019; Zhao et al., 2017; Bolukbasi et al., 2016), 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.

He et al. (2023); Yang et al. (2019) introduce training strategies to use CAMs as labels and refine the classifier's attention toward specific regions. In contrast, Rajabi et al. (2022) proposes transforming the input images to mitigate biases tied to protected attributes like gender. Moreover, Li & Xu (2021) suggests a method to uncover latent biases within image datasets.

Weakly-supervised semantic segmentation (WSSS). (Li et al., 2018) focuses on accurate object segmentation given class labels. The Puzzle-CAM paper (Jo & Yu, 2021) 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 (Sun et al., 2023). Some making use of foundational models such as Segment Anything Model (SAM) (Kirillov et al., 2023) or using multi-modal models like CLIP (Radford et al., 2021).

Priors. Prior knowledge is an important aspect for improving neural network predictions. For example, YOLOv2 (Redmon & Farhadi, 2016) 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 (Zhou et al., 2019; Cai et al., 2020; Hou et al., 2021; Wang & Siddiqi, 2016; Pathak et al., 2015). In particular, Pathak et al. (2015) 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 robustness and localization 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 averaging pooling (GAP) layer (Lin et al., 2014) 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.

For detection and segmentation tasks, GAP is not only applied to the last feature map, but also to the larger feature maps. The intermediate feature maps are part of a pyramid. However, only the last map is usually taken into account for pooling during classification.

This GAP layer can be formally defined as:

$$\mathrm{GAP}(X)_t = \frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m} X_{i,j,t}\,, \tag{1}$$

where X is the last feature map of dimension $n \times m \times c$. For instance, in case of EfficientNet-B0 (Tan & Le, 2020), X has dimension $7 \times 7 \times 1280$ for an input image of size 224×224. The GAP$(\cdot)$ operation reduces X to a vector of size 1280×1. Hence, all of the 7×7 pixels have an effect on the classification. We would like to point out that the term "pixel" here is actually a large area in the original image when considering the receptive field (Luo et al., 2017). For simplicity, we refer to these spatial locations inside the 7×7 matrix as "pixels".

In numerous architectures, there exist attention mechanisms such as the Convolutional Block Attention Module (CBAM) (Woo et al., 2018) or Squeeze-and-Excitation blocks (SE) (Hu et al., 2019) to direct the network's focus towards specific regions. However, these techniques remain incapable of fully constraining the extent of highlighted pixels. For this reason, the GAP layer will still highlight unimportant pixels.

Normally, the features decisive for determining the class do not cover the entirety of the image. For example, when distinguishing between dogs and cats, focusing on the head or even just the eyes can prove to be sufficient. This suggests that a single position in the final low-dimensional feature matrix is often enough to identify the class.

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.

Instead of improving GradCAM, as so many approaches have done before (Chattopadhay et al., 2018; Omeiza et al., 2019; Jiang et al., 2021; Fu et al., 2020; Wang et al., 2020), 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. This makes it possible to incorporate an object size constraint into the model. 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 (Simonyan & Zisserman, 2015), ResNet (He et al., 2015), ConvNeXt (Liu et al., 2022) or EfficientNet (Tan & Le, 2020). Depending on the backbone, we use the last 3 or 4 feature maps as input to a feature pyramid network (FPN) (Lin et al., 2017). 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, a pooling layer such as GAP is employed to obtain a single probability for each class.

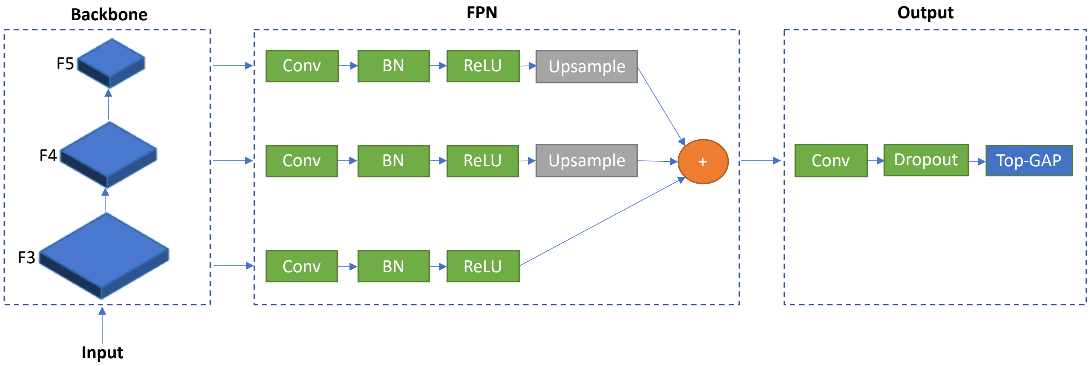


Figure 2: Example of our architecture applied to a backbone with 3 feature maps (e.g. 7 × 7, 14 × 14, 28 × 28). For all convolutions except the final one, a kernel size of 3 and 256 filters is used. The last convolution employs a kernel size of 1, with the number of filters set to match the number of 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 three modes for our model (refer to fig. 2):

1. training: dropout and the pooling layer is enabled
2. prediction: same as training but dropout is deactivated and batch normalization (BN) is frozen
3. CAM: dropout and the pooling layer is deactivated, BN is frozen. The output is normalized to be in the range $[0, 1]$.

The same three modes also exist in standard CNNs. Networks that contain multiple fully-connected layers require the use of GradCAM (Selvaraju et al., 2019) or similar methods. When there is, however, only a single final linear layer, the CAM can be obtained from modifying the last operation. An example shall illustrate this: For EfficientNet-B0, the last feature map has the size $7 \times 7 \times 1280$. This tensor can be reshaped to 49×1280. Then by multiplying this reshaped output with the weight matrix $1280 \times c$ of the original linear layer, we obtain c CAMs with c being the classes.

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 (refer to fig. 2). 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 less parameters than VGG because all additional linear layers are removed.

As indicated in table 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 pixel 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 Jiang et al. (2021), 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 X are considered for averaging. This pooling layer limits the number of pixels that the network can use for generating predictions.

The layer is defined mathematically as follows:

$$\text{Top-GAP}(\tilde{X}, k)_t = \frac{1}{k}\sum_{i=1}^{k} \tilde{X}_{i,t}\,, \tag{2}$$

Here, $\tilde{X}$ represents the ordered feature matrix with dimensions $nm \times c$, where c corresponds to the number of channels. For our model, c is the number of output classes. Each of the c column vectors is arranged in descending order by value, and k values are selected. When $k = 1$, we obtain global max pooling (GMP). When $k = nm$, 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 $\frac{k}{56^2}$ pixels 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 pixels with the highest values, these pixels might not necessarily be the most important ones. Thus, it becomes essential to incentivize the reduction of less important pixels 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:

$$L = \lambda ||X||_1 + \text{CE}\left(\text{softmax}\left(\text{Top-GAP}(\tilde{X}, k)\right), y\right)\,, \tag{3}$$

where $\text{CE}(\hat{y}, y)$ represents the cross-entropy loss between the prediction $\hat{y} = \text{softmax}\left(\text{Top-GAP}(\tilde{X}, k)\right)$ and the ground truth y. Here, λ controls the strength of the regularization. We found that for most datasets $\lambda = 1$ is sufficient.

The main difference between the regular classification loss and our loss is the addition of top-k pooling in conjunction with ℓ_1 regularization.

Lastly, it is important to highlight that it is the combination of these distinct components that yields good results. In our ablation study, we will demonstrate that removing specific components lead to either reduced accuracy or worse feature representations.

4 Evaluation

To demonstrate the generalization capability of our method, we conduct experiments on multiple distinct datasets. We give a detailed description of the datasets in appendix A. These datasets were chosen to have varying characteristics (different class counts, domains and object sizes). We test for each dataset multiple architectures.

We train all models except ImageNet using stratified cross-validation. The results obtained from each fold are then averaged. Employing cross-validation mitigates the impact of randomness on our findings (Picard, 2021).

The main hyperparameter of our approach is given by the pooling layer Top-GAP$(\cdot, k)$. This layer defines the constraint. We always combine this layer with our model structure (see fig. 2) and ℓ_1 loss. We test for all the numerical experiments the values $k \in \{64, 128, 256, 512, 1024, 2048\}$, except for CIFAR-10 where we test $k \in \{8, 16, 32, 64, 128, 256\}$. Due to the high computational cost of training on ImageNet, ResNet-18 was only trained on $k = 256$.

For an image of size 224×224, the output CAM has dimensions 56×56. This means that we use approx. $\frac{64}{56^2} \approx 2\%$ of the feature map for $k = 64$. The highest value that we tested corresponds to $\frac{2048}{56^2} \approx 65\%$.

4.1 Sparsity of the CAMs

Since our approach is concerned with limiting the number of pixels that a neural network can use, we first evaluate sparsity using the ℓ_1 matrix norm. The metric is defined as $\frac{1}{nm}||X||_1$ where $X \in [0, 1]^{n \times m}$ is the CAM (direct output from our model, refer to fig. 2). Although this metric alone does not determine the quality of a CAM, it serves as an indicator of its noise level. In addition, a CAM with fewer highlighted pixels can facilitate the explanation of certain image features.

It is evident from fig. 3 that as we increase the constraint k, the number of displayed pixels on the CAM also rises (i.e. $||X||_1$ rises). This observation validates that our constraint effectively achieves the intended sparsity. While there are some fluctuations for certain datasets and architectures, the overall trend remains consistent.

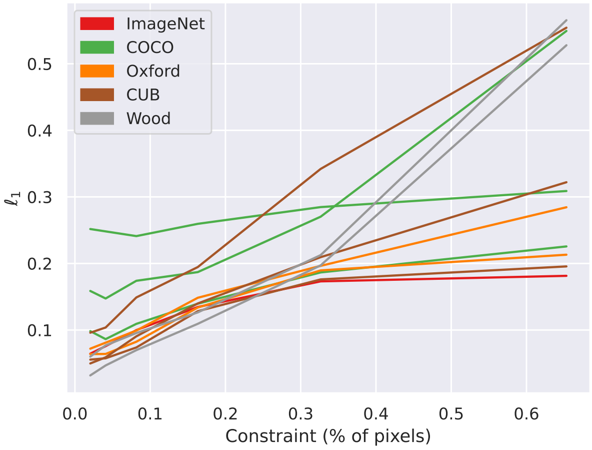


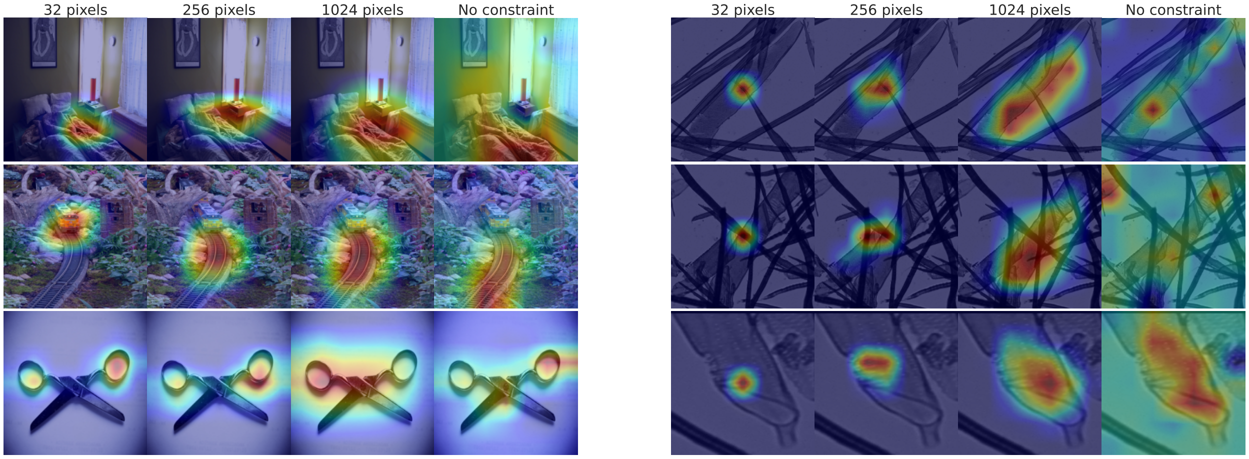
Figure 3: Each line in the graph represents a dataset+architecture combination. The x-axis shows the normalized k value (e.g. $\frac{64}{56^2}$) for the constraint, while the y-axis represents the ℓ_1 norm. The constraint is given by the previously defined pooling layer Top-GAP$(\cdot, k)$.

Details of the results are presented in table 2. 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.

For almost all datasets and architectures, our approach achieved sparser CAMs. We see especially large decreases for ImageNet. To visually demonstrate the effect of the constraint, we use the COCO dataset as an illustrative example.

Dataset	Arch	$\ell_1 \downarrow$	$\ell_1 \downarrow$ (ours)
COCO (Lin et al., 2015)	EN	0.179	**0.064**
COCO (Lin et al., 2015)	CN	0.251	**0.151**
COCO (Lin et al., 2015)	RN	**0.173**	0.194
Wood (Nieradzik et al., 2023)	EN	0.190	**0.032**
Wood (Nieradzik et al., 2023)	CN	0.110	**0.046**
Oxford (Parkhi et al., 2012)	EN	0.154	**0.072**
Oxford (Parkhi et al., 2012)	RN	0.151	**0.064**
CUB-200-2011 (Wah et al., 2011)	EN	0.235	**0.05**
CUB-200-2011 (Wah et al., 2011)	CN	0.164	**0.096**
CUB-200-2011 (Wah et al., 2011)	RN	0.121	**0.056**
ImageNet (Deng et al., 2009)	VG	0.279	**0.064**
ImageNet (Deng et al., 2009)	RN	0.387	**0.123**

Table 2: The last column reports the sparsity of the CAM using our approach (with pixel constraint, ℓ_1 loss and the changes to the model). The third column is a standard model without any changes. For the standard model, we use GradCAM. EN = EfficientNet-B0, CN = ConvNeXt-tiny, RN = ResNet-18, VG = VGG11-BN.



(a) COCO

(b) Wood identification

Figure 4: Impact of pixel constraint on CAM. The pixel constraint is always combined with ℓ_1 loss and our new model structure. "No constraint" denotes a standard unmodified EfficientNet-B0 model using CAM/GradCAM (Selvaraju et al., 2019). For COCO: the ground truth for the first row is "bed". For the second row, it is "train" and for the third row, it is "scissors".

Consider the four images in the second row of fig. 4a showing a train on rails. In the first image, the constraint is 32 pixels and only the train is highlighted. In the 2nd through 4th images, either both the train and the rails or only the rails are highlighted. Since the ground truth class is "train", the rails should not be highlighted because the object ("train") itself best represents the class. With our approach (here: 32 pixels) it is possible to force the network not to use the feature "rails". This shows that our constraint allows to perform bias mitigation.

The first and third row also show that different parts of the objects become more important. When not using any constraint, the CAM is affected by noise. This can be seen in the first row.

Figure 1 and fig. 4b show another example. The wood identification dataset with ConvNeXt-tiny was used for generating fig. 1. Given that the object size can be an important feature, the network attempts to capture this feature even with low constraint values. For instance, with the pixel constraint $k = 64$, we can observe that the network generated dots along the object's edges (see fig. 1a). This suggests that the network is attempting to figure out the object size by employing this strategy.

4.2 Robustness and Accuracy

Limiting the number of pixels accessible to the network does not only impact the CAM but also has an effect on the network's overall functionality. Intuitively, we expect that our approach enhances the network's resilience against adversarial attacks. Typical attacks, like the Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2015), manipulate each pixel uniformly. However, with our constraint, specific pixels in the image no longer exert any influence on predictions. Consequently, we anticipate increased robustness of our network.

To assess this, we perturb the images in the datasets with FGSM and PGD using $\epsilon = {}^{1}/_{255}$ (except for CIFAR-10 where we use $\epsilon = {}^{8}/_{255}$). Then the predicted class of the perturbed images is compared with the class of the original images. The robust accuracy is the percentage of images where the prediction remains the same ("predicted class of perturbed image" equal to "predicted class of original image"). We use the library Foolbox (Rauber et al., 2017; 2020) for the adversarial attacks.

We tested all k and only report the value with the best clean accuracy and robust accuracy (FGSM, PGD).

First, we consider datasets with higher resolution such as ImageNet.

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

Table 3: Our approach refers to the changed model with pixel constraint and ℓ_1 loss. The original models come from PyTorch Image Models (Wightman, 2019) and are pretrained on ImageNet. EN = EfficientNet-B0, CN = ConvNeXt-tiny, RN = ResNet-18, VG = VGG11-bn.

For all the experiments in table 3, we use $\epsilon = {}^{1}/_{255}$ (FGSM/PGD) and 40 steps (PGD). The ± sign denotes the standard deviation of the accuracy across 5 different folds. For ImageNet, we only report a single run due to computational complexity. We achieve a comparable clean accuracy but a much higher adversarial robustness.

Next, we evaluate our method in table 4 on CIFAR-10.

We see a much greater increase in adversarial robustness. Our method comes close to the results of adversarially trained networks while maintaining high accuracy and high training speed.

Overall, we found that there is a strong correlation between robust and clean accuracy. In other words, choosing the k parameter associated with the highest clean accuracy also tends to lead to optimal robustness.

In addition to assessing adversarial robustness, it is valuable to analyze the network's performance under distribution shifts and potential biases. To address this, we use the Waterbirds dataset (Sagawa et al., 2019), where the backgrounds of images are replaced. Furthermore, we evaluate accuracy on ImageNet-Sketch (Wang et al., 2019) and ImageNet-C (Hendrycks & Dietterich, 2019). The results are in table 5.

Method	Arch	PGD^{20} ↑	PGD^{50} ↑	Clean ↑
Baseline	PRN18	0.0	0.0	0.945
Top-GAP (ours)	PRN18	0.517	0.313	**0.951**
FGSM-AT (Andriushchenko & Flammarion, 2020)	PRN18	-	**0.476**	0.81
SAT (Peng et al., 2023)	RN50	**0.552**	-	0.849

Table 4: Results on CIFAR-10. We use $\epsilon = 8/255$ and 20/50 steps. SAT = Standard Adversarial Training, PRN18 = PreAct ResNet-18, RN50 = ResNet-50. Our results are close to the robustness of adversarially trained networks. Refer to appendix B for more experiments.

Dataset	Arch	Acc ↑	Acc ↑ (ours)
CUB → Waterbirds	EN	0.521	**0.564**
CUB → Waterbirds	CN	0.722	**0.737**
CUB → Waterbirds	RN	0.468	**0.52**
ImageNet → Sketch	VG	0.179	**0.20**
ImageNet → Sketch	RN	0.206	**0.236**
ImageNet → ImageNet-C	VG	0.494	**0.498**
ImageNet → ImageNet-C	RN	0.513	**0.535**

Table 5: Evaluation of the out-of-distribution accuracy by using images outside the original dataset. $X \to Y$ means train on X and validate on Y. For instance, we trained an EfficientNet-B0 (EN) model on the CUB dataset and then assessed its accuracy on the Waterbirds dataset. For ImageNet-C (Hendrycks & Dietterich, 2019), we use strength level 1 and take the average of all types of corruption.

4.3 Combination with Adversarial Training

Next, we test whether our approach can be combined with adversarial training. Since adversarial training requires extensive computational resources, only CIFAR-10 is tested.

For our evaluation, we use the RaWideResNet-70-16 model (Peng et al., 2023), which represents the current state-of-the-art on RobustBench. This model was trained with an additional 50 million synthetic images under the (ℓ_∞, $\epsilon = \frac{8}{255}$) threat model. Then we modified this architecture by adding our FPN module, the ℓ_1 loss and the Top-GAP pooling layer. Only the layers of the FPN module were trained, while all other layers were frozen. No adversarial training was used for finetuning. Finally, we compare this modified model with the standard model.

Arch	PGD^{20} ↑	PGD^{50} ↑	Clean Acc ↑
RaWideResNet-70-16	**0.8494**	**0.7462**	0.9372
RaWideResNet-70-16 + ours	0.8463	0.7168	**0.953**

Table 6: The numbers of the standard model differ slightly from the numbers in the original paper because we evaluated all 50,000 images of the test set. We again use $\epsilon = 8/255$.

While we see a slight decrease in robustness of around 3%, the accuracy increases when using our approach. There is an improvement of around 2%.

4.4 Human annotation

Although neural networks may prioritize different regions compared to humans, segmentation masks remain valuable sources of information. For example, if the network focuses on the background instead of the relevant object, it suggests potential classification errors when the object appears no longer with the same background.

The segmentation masks serve as the "ground truth" in our analysis of the COCO and CUB datasets. We compute the pixel-wise intersection over union (IOU) to identify the predicted mask that has the largest overlap with the ground truth. Since standard CNN models do not inherently generate a class activation map, we use the GradCAM method to generate the predicted mask in this context. The results are in table 7.

Dataset	Arch	IOU ↑	IOU ↑ (ours)
COCO	EN	0.309	**0.348**
COCO	CN	0.103	**0.361**
COCO	RN	0.371	**0.391**
CUB	EN	0.323	**0.414**
CUB	CN	0.125	**0.389**
CUB	RN	0.268	**0.435**

Table 7: The last column is our approach ("ours"). The third column is a standard unchanged model. For the standard model, we use GradCAM.

In all cases, our approach consistently demonstrates a higher IOU. Interestingly, ConvNeXt-tiny exhibits a more pronounced improvement ($\approx 25\%$) with our approach compared to the other architectures. The best k value depends here on the actual size of the object. Since the objects of the CUB and COCO datasets are relatively large, we need higher k values.

4.5 Ablation studies

Having established the effectiveness of our approach across various datasets and architectures, our next objective is to assess the impact of the individual components within our solution. We aim to determine whether each component is essential or if certain components can be omitted while still maintaining satisfactory performance. Furthermore, we want to show that the increase of accuracy as seen in table 3 is not a consequence of having a slightly higher number of parameters.

We perform an ablation study on the COCO dataset. There are in total $2^3 = 8$ possibilities as can be seen in table 8.

FPN	ℓ_1 loss	Top-GAP	Acc_{COCO} ↑	IOU_{COCO} ↑	$\text{FGSM}_{\text{COCO}}$
✗	✗	✗	0.801	0.309	0.07
✗	✗	✓	0.681	0.196	0.0
✗	✓	✗	0.799	0.304	0.297
✗	✓	✓	0.796	0.169	0.263
✓	✗	✗	0.796	0.308	0.082
✓	✗	✓	0.532	0.261	0.054
✓	✓	✗	0.790	0.325	0.305
✓	✓	✓	**0.803**	**0.348**	**0.306**

Table 8: Ablation study using the COCO dataset and EfficientNet-B0. The ℓ_1 regularization is beneficial for robustness, but only the combination of all three components also leads to improvements in localization (while maintaining accuracy). We repeated the experiments with the Wood dataset and came to the same results.

When FPN is deactivated, we set the pixel constraint to $k = 4$ for Top-GAP since the final feature matrix has dimensions 7^2. However, when FPN is activated, we increase the constraint to $k = 256$ because the final feature matrix is larger due to upsampling (size 56^2). The ratio is the same e.g. $\frac{1024}{56^2} = \frac{16}{7^2}$.

Table 8 shows that adding a FPN module to the architecture does not consistently increase the accuracy. Only a combination of multiple components leads to an increase.

Although our approach has shown improved accuracy, it is still critical to visually assess the impact of the constraints. To illustrate this, refer to fig. 5. Here we compare all eight configurations using three images from the COCO dataset. The figures clearly show that the combination of the three components: FPN, ℓ_1-loss, and Top-GAP, provides the most favorable results. The figure also shows that the inclusion of some

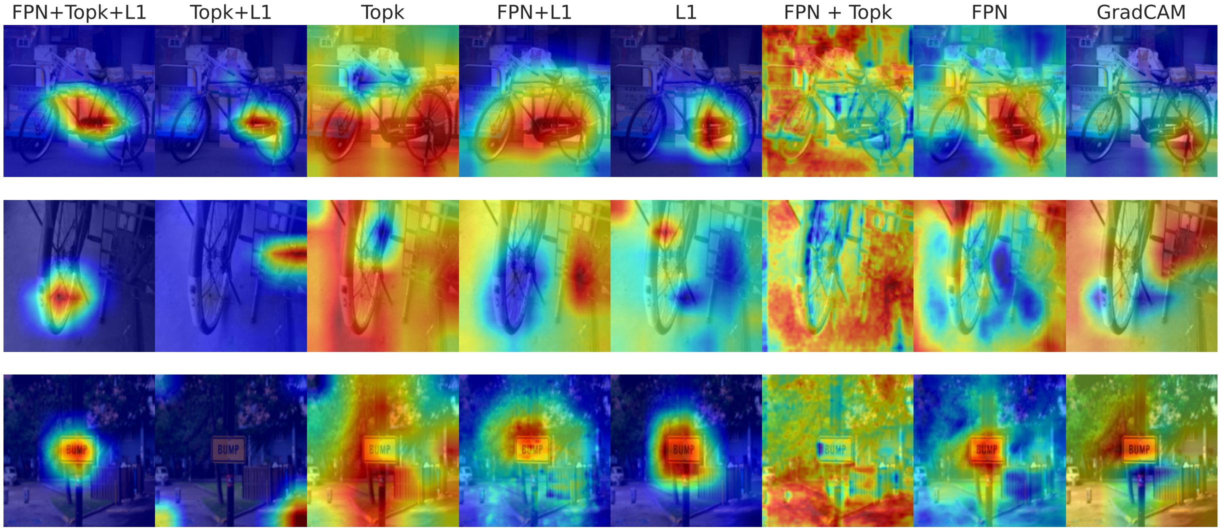


Figure 5: Visual ablation study on COCO. The first two rows show a bicycle, while the third one shows a sign. Only the variant FPN + Top-GAP + ℓ_1 localizes all three objects correctly. The models that were trained with some kind of ℓ_1 regularization tend to have less noise.

form of ℓ_1-loss generally improves the quality of the CAMs by reducing noise. Figure A1 in the appendix shows an example for the wood identification dataset (Nieradzik et al., 2023).

5 Discussion and Outlook

In this paper, we presented a new approach to improve the native robustness of CNNs. Depending on the dataset and architecture, we see major improvements against common adversarial attacks such as FGSM or PGD.

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.

In summary, our approach represents a promising step towards more interpretable and robust CNNs. It opens new possibilities for solving critical problems in computer vision, including robustness, bias mitigation and size priors.

References

Safety and trustworthiness of deep neural networks: A survey. *CoRR*, abs/1812.08342, 2018. URL `http://arxiv.org/abs/1812.08342`. Withdrawn.

Maksym Andriushchenko and Nicolas Flammarion. Understanding and improving fast adversarial training, 2020.

Tolga Bolukbasi, Kai-Wei Chang, James Zou, Venkatesh Saligrama, and Adam Kalai. Man is to computer programmer as woman is to homemaker? debiasing word embeddings, 2016.

Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In Sorelle A. Friedler and Christo Wilson (eds.), *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, volume 81 of *Proceedings of Machine Learning Research*, pp. 77–91. PMLR, 23–24 Feb 2018. URL `https://proceedings.mlr.press/v81/buolamwini18a.html`.

Kaylee Burns, Lisa Anne Hendricks, Kate Saenko, Trevor Darrell, and Anna Rohrbach. Women also snowboard: Overcoming bias in captioning models, 2019.

Jianxiong Cai, Jiawei Hou, Yiren Lu, Hongyu Chen, Laurent Kneip, and Sören Schwertfeger. 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. doi: 10.1109/SSRR50563.2020.9292601.

Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. 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. doi: 10.1109/wacv.2018.00097.

Jacob Clarysse, Julia Hörrmann, and Fanny Yang. Why adversarial training can hurt robust accuracy, 2022.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Ruigang Fu, Qingyong Hu, Xiaohu Dong, Yulan Guo, Yinghui Gao, and Biao Li. Axiom-based grad-cam: Towards accurate visualization and explanation of cnns. *CoRR*, abs/2008.02312, 2020. URL `https://arxiv.org/abs/2008.02312`.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2015.

Sven Gowal, Sylvestre-Alvise Rebuffi, Olivia Wiles, Florian Stimberg, Dan Andrei Calian, and Timothy A. Mann. Improving robustness using generated data. *CoRR*, abs/2110.09468, 2021. URL `https://arxiv.org/abs/2110.09468`.

Julia Grabinski, Steffen Jung, Janis Keuper, and Margret Keuper. Frequencylowcut pooling – plug and play against catastrophic overfitting, 2022.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

Yi He, Xi Yang, Chia-Ming Chang, Haoran Xie, and Takeo Igarashi. Efficient human-in-the-loop system for guiding dnns attention, 2023.

Dan Hendrycks and Thomas G. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *CoRR*, abs/1903.12261, 2019. URL `http://arxiv.org/abs/1903.12261`.

Wei Hou, Xian Tao, and De Xu. Combining prior knowledge with cnn for weak scratch inspection of optical components. *IEEE Transactions on Instrumentation and Measurement*, 70:1–11, 2021. doi: 10.1109/TIM.2020.3011299.

Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. Squeeze-and-excitation networks, 2019.

Hanxun Huang, Yisen Wang, Sarah Monazam Erfani, Quanquan Gu, James Bailey, and Xingjun Ma. Exploring architectural ingredients of adversarially robust deep neural networks, 2022.

Peng-Tao Jiang, Chang-Bin Zhang, Qibin Hou, Ming-Ming Cheng, and Yunchao Wei. Layercam: Exploring hierarchical class activation maps for localization. *IEEE Transactions on Image Processing*, 30:5875–5888, 2021. doi: 10.1109/TIP.2021.3089943.

Sanghyun Jo and In-Jae Yu. Puzzle-CAM: Improved localization via matching partial and full features. In *2021 IEEE International Conference on Image Processing (ICIP)*. IEEE, sep 2021. doi: 10.1109/icip42928.2021.9506058. URL `https://doi.org/10.1109%2Ficip42928.2021.9506058`.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything, 2023.

Kunpeng Li, Ziyan Wu, Kuan-Chuan Peng, Jan Ernst, and Yun Fu. Tell me where to look: Guided attention inference network. *CoRR*, abs/1802.10171, 2018. URL `http://arxiv.org/abs/1802.10171`.

Zhiheng Li and Chenliang Xu. Discover the unknown biased attribute of an image classifier. *CoRR*, abs/2104.14556, 2021. URL `https://arxiv.org/abs/2104.14556`.

Min Lin, Qiang Chen, and Shuicheng Yan. Network in network, 2014.

Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2015.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection, 2017.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s, 2022.

Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. Understanding the effective receptive field in deep convolutional neural networks, 2017.

Lars Nieradzik, Jördis Sieburg-Rockel, Stephanie Helmling, Janis Keuper, Thomas Weibel, Andrea Olbrich, and Henrike Stephani. Automating wood species detection and classification in microscopic images of fibrous materials with deep learning, 2023.

Daniel Omeiza, Skyler Speakman, Celia Cintas, and Komminist Weldemariam. Smooth grad-cam++: An enhanced inference level visualization technique for deep convolutional neural network models. *CoRR*, abs/1908.01224, 2019. URL `http://arxiv.org/abs/1908.01224`.

O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar. Cats and dogs. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

Deepak Pathak, Philipp Krähenbühl, and Trevor Darrell. Constrained convolutional neural networks for weakly supervised segmentation, 2015.

ShengYun Peng, Weilin Xu, Cory Cornelius, Matthew Hull, Kevin Li, Rahul Duggal, Mansi Phute, Jason Martin, and Duen Horng Chau. Robust principles: Architectural design principles for adversarially robust cnns, 2023.

David Picard. Torch.manual_seed(3407) is all you need: On the influence of random seeds in deep learning architectures for computer vision. *CoRR*, abs/2109.08203, 2021. URL `https://arxiv.org/abs/2109.08203`.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John C. Duchi, and Percy Liang. Adversarial training can hurt generalization. *CoRR*, abs/1906.06032, 2019. URL `http://arxiv.org/abs/1906.06032`.

Amirarsalan Rajabi, Mehdi Yazdani-Jahromi, Ozlem Ozmen Garibay, and Gita Sukthankar. Through a fair looking-glass: mitigating bias in image datasets, 2022.

Jonas Rauber, Wieland Brendel, and Matthias Bethge. Foolbox: A python toolbox to benchmark the robustness of machine learning models. In *Reliable Machine Learning in the Wild Workshop, 34th International Conference on Machine Learning*, 2017. URL `http://arxiv.org/abs/1707.04131`.

Jonas Rauber, Roland Zimmermann, Matthias Bethge, and Wieland Brendel. Foolbox native: Fast adversarial attacks to benchmark the robustness of machine learning models in pytorch, tensorflow, and jax. *Journal of Open Source Software*, 5(53):2607, 2020. doi: 10.21105/joss.02607. URL `https://doi.org/10.21105/joss.02607`.

Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger, 2016.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier, 2016.

Shiori Sagawa, Pang Wei Koh, Tatsunori B. Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *CoRR*, abs/1911.08731, 2019. URL `http://arxiv.org/abs/1911.08731`.

Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 128(2):336–359, oct 2019. doi: 10.1007/s11263-019-01228-7. URL `https://doi.org/10.1007%2Fs11263-019-01228-7`.

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

Weixuan Sun, Zheyuan Liu, Yanhao Zhang, Yiran Zhong, and Nick Barnes. An alternative to wsss? an empirical study of the segment anything model (sam) on weakly-supervised semantic segmentation problems, 2023.

Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2020.

Florian Tramèr and Dan Boneh. Adversarial training and robustness for multiple perturbations, 2019.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. *The Caltech-UCSD Birds-200-2011 Dataset.* Jul 2011.

Chu Wang and Kaleem Siddiqi. 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. doi: 10.1109/CVPRW.2016.130.

Haofan Wang, Zifan Wang, Mengnan Du, Fan Yang, Zijian Zhang, Sirui Ding, Piotr Mardziel, and Xia Hu. Score-cam: Score-weighted visual explanations for convolutional neural networks, 2020.

Haohan Wang, Songwei Ge, Eric P. Xing, and Zachary C. Lipton. Learning robust global representations by penalizing local predictive power. *CoRR*, abs/1905.13549, 2019. URL `http://arxiv.org/abs/1905.13549`.

Zekai Wang, Tianyu Pang, Chao Du, Min Lin, Weiwei Liu, and Shuicheng Yan. Better diffusion models further improve adversarial training, 2023.

Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019.

Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module, 2018.

Xi Yang, Bojian Wu, Issei Sato, and Takeo Igarashi. Directing dnns attention for facial attribution classification using gradient-weighted class activation mapping. *CoRR*, abs/1905.00593, 2019. URL `http://arxiv.org/abs/1905.00593`.

Ákos Zarándy, Csaba Rekeczky, Péter Szolgay, and Leon O Chua. 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.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 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, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1323. URL `https://aclanthology.org/D17-1323`.

X. Zhou, M. Zhu, G. Pavlakos, S. Leonardos, K. G. Derpanis, and K. Daniilidis. 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. ISSN 1939-3539. doi: 10.1109/TPAMI.2018.2816031.

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

Supplementary Material

The appendix contains the following additional materials:

- A detailed description of the datasets.
- A visual ablation study for the wood dataset: fig. A1
- More experiments for the CIFAR datasets: appendix B

A Description of datasets

We test all our models on the following datasets:

- COCO (Lin et al., 2015): 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 (Nieradzik et al., 2023): This dataset consists of high-resolution microscopy images for hardwood fiber material. Nine distinct wood species have to be distinguished.
- Oxford-IIIT Pet Dataset (Parkhi et al., 2012): The task is to differentiate among 37 breeds of dogs and cats.
- CUB-200-2011 (Wah et al., 2011) and Waterbirds (Sagawa et al., 2019): 200 classes of birds have to be distinguished. Waterbirds replaces the background of the original images to test the models for biases.
- ImageNet (Deng et al., 2009): A large-scale dataset with 1000 different classes. ImageNet-Sketch (Wang et al., 2019) / ImageNet-C (Hendrycks & Dietterich, 2019) 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.

A.1 Visual ablation study

We compare three configurations: ℓ_1 + Top-GAP$(\tilde{X}, k)$, FPN + ℓ_1 + Top-GAP$(\tilde{X}, k)$ and no constraint (standard EfficientNet-B0 model)

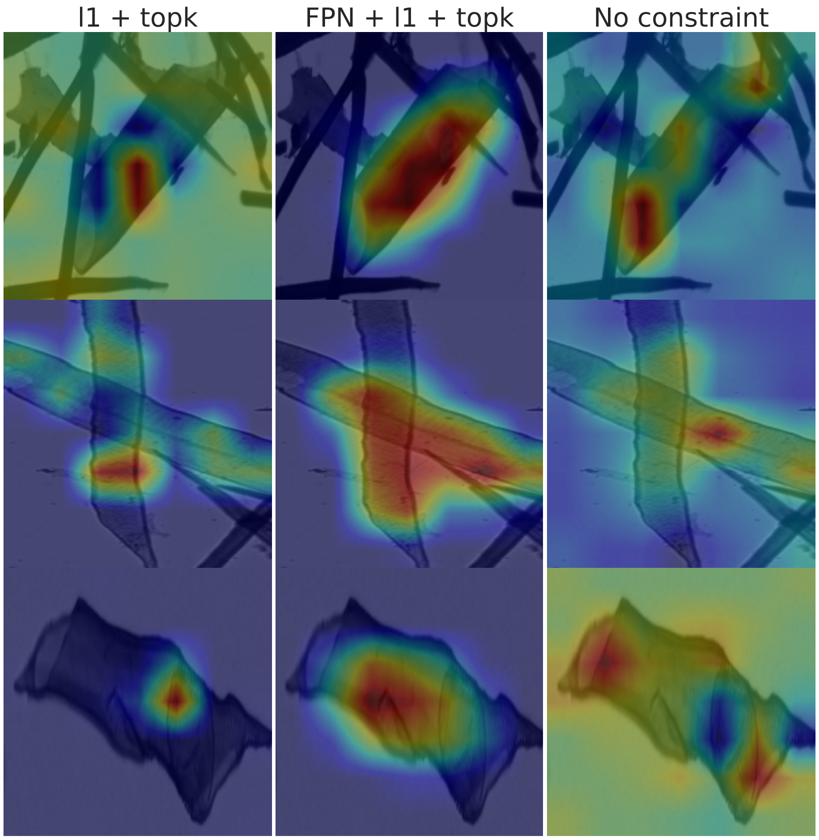


Figure A1: Ablation study on the wood identification study with three different configurations.

The variant without constraint shows considerably more noise.

B More CIFAR-10 experiments

B.1 Effect of steps on PGD

We analyze the effect of the parameter `steps` on the function `LinfProjectedGradientDescentAttack` of Foolbox. Three models are compared: Andriushchenko2020Understanding (Andriushchenko & Flammarion, 2020), baseline (standard PreAct ResNet-18), Top-GAP (our approach). We sampled 500 images from the dataset.

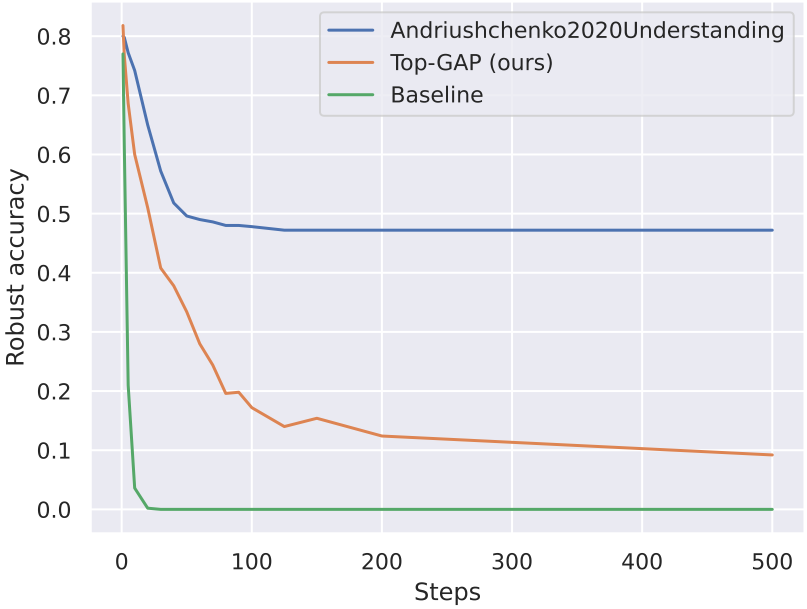


Figure A2: Effect of steps on ℓ_∞-PGD

B.2 Architectures

The experiments from the main paper were repeated for different architectures. Furthermore, the hyperparameters were optimized for robust accuracy. The results were averaged across 5 different seeds.

Arch	PGD^{20} ↑	PGD^{50} ↑	Clean Acc ↑
PreAct-ResNet18	**0.5484** ± 0.0135	**0.3528** ± 0.0277	0.9493 ± 0.0013
ResNet18	0.5407 ± 0.033	0.3361 ± 0.0516	0.9501 ± 0.0009
ResNet34	0.4254 ± 0.08	0.2989 ± 0.1181	0.9513 ± 0.0023
ResNet50	0.4815 ± 0.0299	0.336 ± 0.0462	**0.9515** ± 0.0023
WideResNet40-4	0.4156 ± 0.0092	0.2811 ± 0.0253	0.9462 ± 0.0002

Table A1: We use $\epsilon = {}^{8}/_{255}$ and 20/50 steps.