

# Learning Interpretable Concept Groups in CNNs

Saurabh Varshneya<sup>1\*</sup>, Antoine Ledent<sup>1</sup>, Robert A. Vandermeulen<sup>2</sup>, Yunwen Lei<sup>3</sup>, Matthias Enders<sup>4</sup>, Damian Borth<sup>5</sup> and Marius Kloft<sup>1</sup>

<sup>1</sup>Technical University of Kaiserslautern, Germany

<sup>2</sup>Technical University of Berlin, Germany

<sup>3</sup>University of Birmingham, United Kingdom

<sup>4</sup>NPZ Innovation GmbH, Germany

<sup>5</sup>University of St.Gallen, Switzerland

{varshneya, ledent, kloft}@cs.uni-kl.de, vandermeulen@tu-berlin.de, y.lei@bham.ac.uk, m.enders@npz-innovation.de, damian.borth@unisg.ch

## Abstract

We propose a novel training methodology—Concept Group Learning (CGL)—that encourages training of interpretable CNN filters by partitioning filters in each layer into *concept groups*, each of which is trained to learn a single visual concept. We achieve this through a novel regularization strategy that forces filters in the same group to be active in similar image regions for a given layer. We additionally use a regularizer to encourage a sparse weighting of the concept groups in each layer so that a few concept groups can have greater importance than others. We quantitatively evaluate CGL’s model interpretability using standard interpretability evaluation techniques and find that our method increases interpretability scores in most cases. Qualitatively we compare the image regions that are most active under filters learned using CGL versus filters learned without CGL and find that CGL activation regions more strongly concentrate around semantically relevant features.

There exists much work that focuses on understanding the hidden interpretations of CNNs, but there is a lack of work that actively influences the learning process to favor interpretable representations.

To address this gap we propose a framework for CNNs that not only *assesses* interpretability but *induces* it. Our approach is to induce a group structure in CNNs at training, where groups of filters with a useful semantic meaning emerge during the training process. We achieve this through carefully chosen regularization and auxiliary loss functions. In our method, before any training is performed, we partition the collection of filters in each layer into groups, which we term *concept groups*. During training we promote filters within the same concept group to learn similar features, so that, after training, they form a group of correlated filters that jointly encode an abstract visual concept (e.g., an object, scene, or color).

To learn these concept groups, we introduce three novelties to our training objective (illustrated in Figure 1): 1. We propose a new loss function (entitled *group activation*

## 1 Introduction

There is a great interest in understanding the hidden representations produced by convolutional neural networks (CNNs). Understanding these representations has significant implications theoretically (a better understanding of neural networks in general) and practically (trustworthiness in AI). Toward this end, many methods for interpreting hidden representations have been proposed. We delineate two avenues of research, which include (1) visualizing the regions of images that lead to high activation responses in a filter [Yosinski *et al.*, 2014; Zhang *et al.*, 2017; Mahendran and Vedaldi, 2016] and (2) interpreting the semantic meaning learned by a filter by finding its alignment to human-interpretable visual concepts, such as object, scene, and color [Bau *et al.*, 2017; Zhou *et al.*, 2018; Zhang *et al.*, 2020].

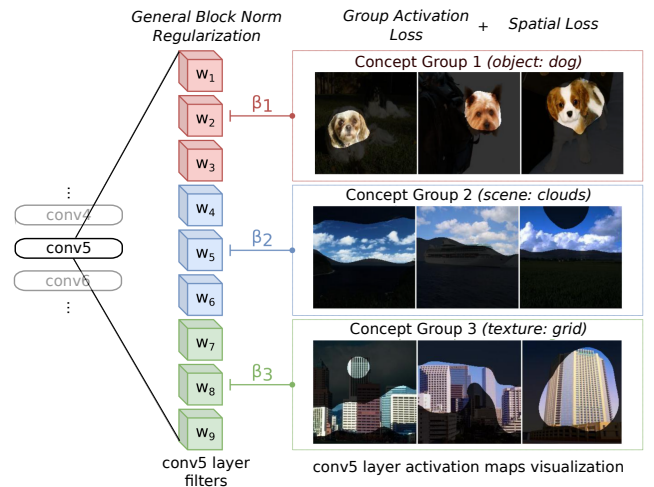


Figure 1: An illustration of our framework of learning concept groups in a convolutional layer of a CNN.

\*Contact Author: varshneya@cs.uni-kl.de

Code will be available at: <https://github.com/srb-cv/cgl>

*loss*) that encourages the filters within each concept group to have highly overlapping activation regions for a given sample. 2. We propose another loss function (entitled *spatial loss*), which promotes the activations of each filter to be concentrated around their mean position. 3. Using an  $\ell_{2,1}$ -block-norm regularizer on the concept groups, we learn a sparse weighting of the concept groups. These weights can be interpreted as the relative importance of each concept group. A key property of our method is that it is completely unsupervised, automatically finding concepts using zero side-information. In practice one would likely include our losses in conjunction with some other task, e.g. classification, to increase interpretability of that task, but none of the additional task information, e.g. class labels, is used in our losses.

In addition to the training method described above, it can be observed that interpretability of CNNs decreases when batch-normalization layers are introduced [Zhou *et al.*, 2018]. To remedy this we incorporate a variation of batch-norm which works together with its regularizer and losses to achieve the desired group structure in the hidden representations and enhance interpretability.

We perform a quantitative analysis following the experimental setup by [Bau *et al.*, 2017] and find that our method significantly improves interpretability. We also analyze the filters of our model qualitatively by visualizing the filters’ activations and find that our training setup yields representations with noticeably more interpretable filters.

## 2 Related Work

We mention here a few existing works that are related to our training method and outline the major differences with the existing approaches.

**Group convolutions and structured sparsity:** Group convolutions were first used in Alexnet [Krizhevsky *et al.*, 2017] by distributing filters into groups to train the model over multiple GPUs. It is important to note that our concept group approach is different from group convolutions, where there is no connection among the activation maps of different partitioned groups. In contrast, concept groups’ activation maps are concatenated at each layer before feeding them to the next layer (see Figure 3).

There exist methods that induce structured sparsity in CNNs [Wen *et al.*, 2016; Li *et al.*, 2017]. Such methods show that inducing sparsity among groups of filters can achieve state-of-the-art accuracy on various datasets with reduced computational costs for CNNs. However, less is known about the effect of such a regularization on the interpretability of networks.

**Disentangled representations:** For improving hidden representations while training, a group of methods endeavours to obtain *disentangled representations*. The aim of such methods coincides with the main focus of our research. These methods aim to learn better representations where each filter represents a clear and unique visual concept. [Zhang *et al.*, 2018] introduce a training method which aims to train each filter in the final convolutional layers of a CNN to represent an object part. The activation loss from their method encour-

ages a filter to activate only for the training images that belong to the assigned category in the training set.

Our own techniques also rely on applying a loss to the activation maps of filters to inject priors into the CNN representation. However, there is a major difference in how we apply the losses on activations in our study. In [Zhang *et al.*, 2018] each filter is pushed towards a *known* concept corresponding to an object part. Ultimately the *concepts* that this method extracts are parts of the objects in the training classes. This is unlike our method which can capture *arbitrary* concepts (not only ones corresponding to object parts) and without any supervision. Furthermore, the method from [Zhang *et al.*, 2018] filters out activations of low magnitude entirely in the forward pass. We argue that such hard constraints on the activation map can hamper the representation power of the network. To retain representation power our method applies soft constraints on the activations. This does not require any changes in the forward pass.

The rest of this paper is organized as follows: in Section 3, we mathematically define our algorithm and precise regularization procedure to make the CNNs more interpretable. In Section 4.1, we explain the evaluation techniques by [Bau *et al.*, 2017], which we later use to evaluate the interpretability of the networks trained with our own training methods. Finally, in Section 4.3, we compare the results of the experiments we performed with the existing state-of-the-art methods.

## 3 Algorithm

Here we present our method which induces interpretable filters in all layers of a CNN. We briefly outline our method here and describe them precisely in the sequel. Central to our method is an initial grouping of each convolutional layer’s filters into  $G$  equal-size groups, e.g. for a collection of filters in a layer indexed by  $1, \dots, F$  we divide the indices into  $G$  groups  $(1, \dots, f), (f+1, \dots, 2f), \dots, ((G-1)f+1, \dots, Gf)$  with  $F = Gf$ . During training we aim to have a single group of filters (e.g.  $(f+1, \dots, 2f)$ ) correspond to a single concept. We assign multiple filters to the same concept in the hopes of robustly capturing the concept in a way that would not be possible with a single filter. The number of groups is a hyperparameter that is determined before training. To make the method less restrictive, we can leave some “free” filters outside the grouping. We consider each concept as a block matrix and apply a general block norm over the collection of groups so as to encourage structured sparsity [Wen *et al.*, 2016]. The filters belonging to each group are trained to correspond to similar visual concepts. To achieve this we feed each filter’s activation map through a sigmoid to determine a (soft) receptive field which represents the areas of the image where the filter is active and then apply a loss to induce filters in the same concept group to have receptive fields that largely overlap and are concentrated to a connected region within an image (illustrated in Figure 2). We now precisely describe the regularizers in the following.

### 3.1 Group Activation Loss $L_g$

Here we describe our loss function that encourages filters in the same concept group to learn similar concepts. To do this

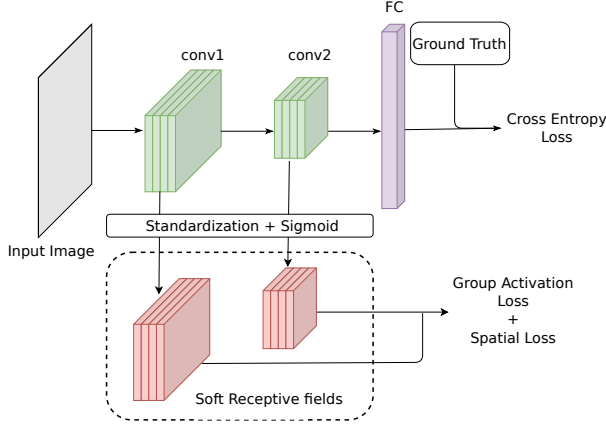


Figure 2: The training setup for our auxiliary loss functions. At every layer we obtain a soft receptive field by feeding a suitably standardized version of the activation maps through a sigmoid. We apply constraints on the soft receptive fields by our well-defined losses

our loss function, which we denote by  $L_g$ , will encourage filters within the same concept group to be active in roughly the same area for each input image. To determine whether a filter is “active” in a region of the image, we first feed the activation maps before applying any non-linearity (we call this the *pre-activation map*) through a linear scaling (which is also learned and shared between layers) followed by a sigmoid. We interpret the output of the sigmoid as the probability of a filter being active at a location.

Let us denote the activation map of the  $i^{\text{th}}$  filter in group  $g$  in layer  $l$  by  $a_{ig}^l \in \mathbb{R}^{A_l}$ . Here,  $A_l$  denotes the size of the pre-activation maps in layer  $l$ . A filter and its activation maps are typically tensors but we simply consider a flattened version of them for notational simplicity.

From our pre-activation map we utilize a sigmoid to yield a *soft-receptive field*,  $\psi_{ig}^l \in (0, 1)^{A_l}$ , which is defined by the following:

$$\psi_{ig}^l = \sigma \left( P_1 \frac{a_{ig}^l}{S_{ig}^l} + P_2 \right), \quad (1)$$

where  $S_{ig}^l$  is the standard deviation of the  $i^{\text{th}}$  activation-map of group  $g$ , computed over the minibatch. Note that  $P_1$  and  $P_2$  are a pair of scaling parameters which can be learned during training. These parameters are shared for all the activation maps in the network. We now define the distance between two soft-receptive fields  $\psi_1$  and  $\psi_2$  as

$$\mathcal{D}(\psi_1, \psi_2) = \frac{2\|\psi_1 - \psi_2\|_1}{\|\psi_1\|_1 + \|\psi_2\|_1 + \|\psi_2 - \psi_1\|_1}, \quad (2)$$

where  $\|\cdot\|_1$  denotes the  $L^1$  norm:  $\|x\|_1 = \sum_i |x_i|$ .

The motivation of this distance is to create a soft equivalent to one minus the IoU (intersection over union). This score is frequently used in computer vision to assess the performance of tracking or localization algorithms [Tychsen-Smith and Petersson, 2018; Huang *et al.*, 2019]. To see the equivalence, suppose that  $\psi_1$  and  $\psi_2$  are the indicator functions of two sets

$A$  and  $B$ . Then we have

$$\begin{aligned} \mathcal{D}(\psi_1, \psi_2) &= \frac{2\|\psi_1 - \psi_2\|_1}{\|\psi_1\|_1 + \|\psi_2\|_1 + \|\psi_1 - \psi_2\|_1} \\ &= \frac{2\mu(A \Delta B)}{\mu(A) + \mu(B) + \mu(A \Delta B)} = \frac{2\mu(A \Delta B)}{2\mu(A \cup B)} \\ &= 1 - \frac{\mu(A \cap B)}{\mu(A \cup B)} = 1 - \text{IOU}, \end{aligned} \quad (3)$$

where we use the standard notation  $A \Delta B := (A \setminus B) \cup (B \setminus A) = (A \cup B) \setminus (A \cap B)$  for the symmetric difference between  $A$  and  $B$ , and  $\mu(\cdot)$  denotes the Lebesgue measure or the cardinality in the case of finite sets. The definition of  $\psi$  must be altered slightly to work with batch normalization; we describe this in the following.

### Definition of $\psi$ With Batch-normalization

If the network is trained with batch-norm, the pre-activations are standardized to have mean 0 and variance 1. The input to the next layer is then  $\gamma_{ig}^l(a_{ig}^l) + \beta_{ig}^l$ , where  $\beta_{ig}^l$  and  $\gamma_{ig}^l$  are the learned parameters of a batch-norm layer over the  $i^{\text{th}}$  activation-map, of group  $g$ , as in standard batch normalization [Ioffe and Szegedy, 2015]. To modify the batch normalized values to be compatible with our method we first define a variable  $\tau_{ig}^l$  by

$$\tau_{ig}^l = a_{ig}^l + \frac{(\beta_{ig}^l)}{(\gamma_{ig}^l)},$$

which can be thought of as undoing batch normalization for the purposes of finding a soft receptive field, since batch normalization is known to reduce interpretability [Bau *et al.*, 2017]. We can now feed  $\tau_{ig}^l$  through a sigmoid to obtain our soft receptive field as follows:

$$\psi_{ig}^l = \sigma(P_1 \tau_{ig}^l + P_2). \quad (4)$$

### Formula for $L_g$

We can now define our group activation loss  $L_g$  to be the following (note that the second term is new and is described in the next paragraph)

$$\begin{aligned} &\frac{1}{r} \sum_{l,g} \sum_{(i,j) \in \mathcal{R}} \frac{2\|\psi_{jg}^l - \psi_{ig}^l\|_1}{\|\psi_{jg}^l\|_1 + \|\psi_{ig}^l\|_1 + \|\psi_{jg}^l - \psi_{ig}^l\|_1} + \dots \\ &\lambda_2 \frac{1}{r} \sum_{l,g} \sum_{(i,j) \in \mathcal{R}} \frac{2\|\psi_{jg}^l - \psi_{ig}^{l+1}\|_1}{\|\psi_{jg}^l\|_1 + \|\psi_{ig}^{l+1}\|_1 + \|\psi_{jg}^l - \psi_{ig}^{l+1}\|_1} \end{aligned} \quad (5)$$

where  $\mathcal{R}$  is a random subset of cardinality  $r$  of  $\{1, 2, \dots, N_g^l\} \times \{1, 2, \dots, N_g^l\}$  that changes at each iteration of the gradient descent procedure.  $N_g^l$  denotes the number of activation maps in group  $g$  in layer  $l$ . We use this randomization over  $\mathcal{R}$  as an approximation of comparing all pairs of filters in each concept group which is computationally expensive. Here we write  $\psi_{jg}^l$  for the vector composed of the components  $\psi_{jg}^l(x_i)$  for samples  $x_i$  in some minibatch. All the  $L^1$  norms are computed over the minibatch (as well as the spatial dimension). The value for  $r$  can be set as a hyperparameter. We find that setting  $r$  to  $3N_g^l$  works well in practice.

Each term in the summation that appears in  $L_g$  is a soft version of one minus the intersection over union measure of similarity between the receptive fields given by  $\mathcal{D}(\psi_{ig}^l, \psi_{jg}^l)$ . As some concepts typically gradually change or die out in higher layers, we additionally consider employing a sliding comparison, the second term in (5), where the filters in group  $g$  of layer  $l$  are required to be similar to those in group  $g$  of layer  $l + 1$  but are not directly compared to those in group  $g$  in layers beyond  $l + 1$ . We found that enforcing this similarity of concepts throughout layers is too strong a requirement and simply causes the filters to go to zero. Because of this we use  $\lambda_2 = 0$  for all  $l$  in our experiments.

### 3.2 Spatial Loss $L_s$

In this section we introduce a loss that aims to regularize each filter to activate on a single, connected area of each image. Suppose for instance that an image contains a car and a house both of which are relevant concepts to the task to which the CNN is being applied. Since both concepts are abstract they are likely to be represented near the final convolutional layers of a CNN where the receptive fields are large. We would like to keep these concepts separate between concept groups and prevent them from incorporating large portions of the image corresponding to multiple abstract concepts. To achieve this we enforce each filter to only be active on a small connected region on images by penalizing activation responses that far from the center of activation. The center of activation is computed over the soft receptive field  $\psi$  for each activation map. We define spatial loss  $L_s(\psi)$  as the following:

$$L_s(\psi) = \frac{\sum_j \psi_j \|j - c(\psi)\|_2}{\sum_j \psi_j}, \text{ where } c(\psi) = \frac{\sum_j j \cdot \psi_j}{\sum_j \psi_j}$$

and  $j$  denotes the index of a activation of the soft receptive field ( $\psi$ ) of a filter. Here  $c$  represents the mean position of activation of a soft receptive field and  $L_s$  is analogous to the corresponding variance. The loss is computed over all the activation maps of a filter obtained in a minibatch.

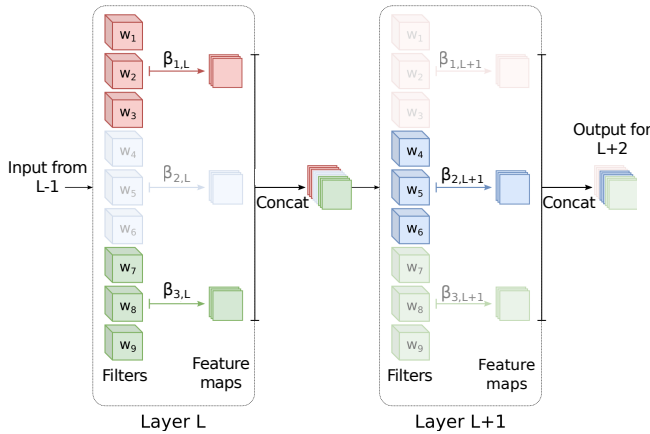


Figure 3: The general block norm induces a group structure in the filters of a layer, where few groups are dominantly learned over other groups.  $\beta_1$ ,  $\beta_2$  and  $\beta_3$  are the relevance factors of each group which are learned while training a convolutional neural network (CNN)

### 3.3 General Block Norm Regularizer $R_{bn}$

In this section we define our general block norm regularizer  $R_{bn}$  that replaces the conventional weight decay regularizer. This changes the inductive bias of our training algorithm so that it induces a group sparse structure over the concept groups. This regularizer encourages some concept groups to be of greater importance than others, thereby giving a notion of importance to concept groups and encouraging our concept groups to be parsimonious. This allows us to avoid situations where, for example, multiple concept groups capture the concept of “car.” Such sparse models also yield reduced memory requirements and improved training speed as side benefits [Mocanu *et al.*, 2018]. An illustration the result of using this regularization is shown in Figure 3. We will denote by  $w_{gl}$  the matrix containing all the weights of the filters of group  $g$  in layer  $l$ . The regularizer  $R_{bn}$  can then be written as :

$$R_{bn} = \sum_{l=1}^L \sum_{g=1}^G \|w_{gl}\|_{Fr}, \quad (6)$$

where  $L$  denotes the number of convolutional layers in the network,  $G$  the number of concept groups per convolutional layer, and  $\|\cdot\|_{Fr}$  denotes the Frobenius norm i.e.,  $\|M\|_{Fr}^2 = \sum_{i,j} M_{i,j}^2$  for a matrix  $M$ .

An advantage of using the general block norm regularizer  $R_{bn}$  is that we can obtain a vector of relevance factors  $\beta$  for our concept groups, as shown in Figure 3. These relevance scores can be obtained by a summation of the norms as follows:

$$\beta_{gl} = \|w_{gl}\|_{Fr} \quad \text{and} \quad \beta_l = \sum_{g=1}^G \beta_{gl}, \quad (7)$$

where,  $\beta_{gl}$  denotes a group’s relevance in layer  $l$ , and  $\beta_l$  denotes the relevance of layer  $l$  in a network.

**Complete framework:** In summary our overall framework is obtained by combining the above mentioned regularization strategy and the two loss functions. For a simplified view of the combined method, we consider a CNN with weight parameters  $w$  and we denote the soft receptive fields obtained from the activation maps by  $\psi$ . The final optimization target for an interpretable CNN can be defined as:

$$L = L_d(w) + \lambda_{bn} R_{bn}(w) + \lambda_g L_g(\psi) + \lambda_s L_s(\psi), \quad (8)$$

where,  $L_d$  denotes the main loss for our task (e.g. cross entropy for classification),  $R_{bn}$  is the general block norm regularization,  $L_g$  and  $L_s$  are the group activation loss and spatial loss respectively.  $\lambda_{bn}$ ,  $\lambda_g$  and  $\lambda_s$  are the weightings of the defined regularization and auxiliary losses with respect to the main loss.

## 4 Experiments

In this section we explain the evaluation strategy we adopt for the quantitative and qualitative analysis of the interpretability of CNNs, and describe the results of our experiments performed on synthetic and real world datasets.

#### 4.1 Interpretability: Quantitative Assessment

In order to quantify the interpretability of a trained CNN, we exactly replicate the evaluation on the Broden dataset as proposed by [Bau *et al.*, 2017]. The Broden dataset contains pixel-wise annotations of a broad range of categories, which belong to one of six human-interpretable visual concepts. We compute the alignment of a filter with a category by comparing its activation maps for a set of images against the available ground truth using the same threshold and scoring function described by [Bau *et al.*, 2017]. The interpretability of a layer can then be defined as the number of unique categories assigned to the filters in that layer. Authors call this as the number of *unique detectors*.

##### Evaluation on a Synthetic Dataset

In order to validate the efficacy of our methods in a controlled environment, we constructed a synthetic dataset with obvious human-interpretable visual concepts corresponding to shape and color: each image contains two randomly positioned figures among the set {square, circle, triangle} with repetitions allowed, and each shape is assigned a colour from the set {red, blue, green}. Note that in principle, there are 15 high-level concepts in this paradigm: one for each combination of colour and shape, and one for each colour and each shape separately.

We consider two training settings: in the first case, we have one label for each combination of shapes and colours (there are therefore 45 labels). In the second setting, we simply assign the label one if a square is present and zero otherwise (the problem is therefore binary). Therefore, in the first setting, all the information contained in the high level concepts “shape, color, color-shape” is contained in the label, whereas in the second situation, only part of this information is encoded in the label.

For training, we use a simple CNN with 2 convolutional layers, having 128 and 256 filters in the first and second layer respectively. We set the size of filters to  $3 \times 3$  for both the layers. The evaluation is performed using the same metrics as described above. We report the results of the second, binary setting in Table 1. In the multiclass setting, the network is able to recover all concepts very well regardless of regularization. However, we find that in the binary setting, the addition of our regularizers has a very strong positive effect on the recovery of the concepts: in the case where we add all three regularizers, 14 out of 15 of all unique concepts are recovered at the second layer, compared to only 10 when using weight decay.

This is a significant illustration of the fact that our method is able to guide the network towards a more complete and disentangled representation of all salient features of the data in the absence of low-level feature annotations.

##### Evaluation on the Real Datasets

We tested our methods on three well-known convolutional architectures named Alexnet, Alexnet-B and VGG, where Alexnet-B is the version containing the batch normalization layers added to the common Alexnet. All the networks are trained from scratch on the two well-known image classification datasets Places365 [Zhou *et al.*, 2017] and Imagenet [Deng *et al.*, 2009].

Regularizer	layer	No. of Unique Detectors			
		color	shape	color-shape	Total
Weight-Decay	conv1	3	0	4	7
	conv2	3	0	7	10
$R_{bn}$	conv1	3	2	4	9
	conv2	3	2	7	12
$R_{bn} + L_g + L_s$	conv1	3	3	5	11
	conv2	3	2	9	14

Table 1: Number of unique detectors in each layer while training on synthetic data in a two class labels setup

geNet [Deng *et al.*, 2009]. The Places365 dataset is comparable to the ImageNet dataset, which contains 1.8 million training images with 365 scene categories. For evaluation, we exactly replicate the method from [Bau *et al.*, 2017], as explained above to compute the interpretability of the trained models. For a baseline of interpretability, we obtained the pretrained models for Imagenet and Places365 datasets from the Pytorch and MIT CSAIL repositories respectively. Table 2 shows the comparison of the interpretability scores for the models trained with the conventional weight decay regularizer and our training method.

##### Analysis over Concept-groups

For each predefined group, we compute a weighted sum over the number of detectors found in the group and the IoU score. A group is said to be aligned to a visual concept when the weighted sum is greater than a threshold  $T$ , which was chosen by hand and is the same for all experiments. One can obtain the relevance of such a concept-group from Equation 7. In Figure 4, we show an analysis over the emergent concept-groups in the convolutional layers of Alexnet.

#### 4.2 Interpretability: Qualitative Assessment

We use the method described in [Zhou *et al.*, 2014] for visualizing filters of trained CNNs. To visualize the receptive

Dataset	Model	conv1	conv2	conv3	conv4	conv5	score
Places365	Alexnet	6	12	22	36	54	130
	Alexnet, ( $R_{bn}$ )	6	16	36	46	69	173
	Alexnet, ( $R_{bn} L_g L_s$ )	8	17	35	50	75	185
Places365	Alexnet-B	6	13	27	43	59	148
	Alexnet-B, ( $R_{bn}$ )	6	15	30	44	62	157
	Alexnet-B, ( $R_{bn} L_g L_s$ )	5	17	30	51	63	166
Imagenet	Alexnet	5	20	34	28	49	136
	Alexnet, ( $R_{bn}$ )	5	16	39	32	45	137
	Alexnet, ( $R_{bn} L_g L_s$ )	7	16	34	36	48	141
conv3-3 conv4-3 conv5-1 conv5-2 conv5-3							
Imagenet	VGG16	10	48	59	53	89	259
	VGG16, ( $R_{bn} L_g L_s$ )	14	48	59	62	83	266
Places365	VGG16	9	50	62	75	119	315
	VGG16, ( $R_{bn} L_g L_s$ )	10	45	62	86	129	332

Table 2: The table above shows the sum of the number of unique detectors in each layer of the network. We achieve a better interpretability across all the tested networks and datasets.



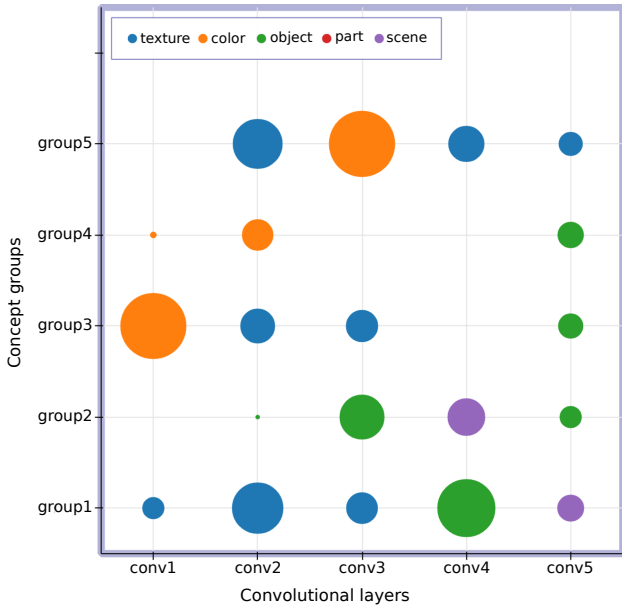


Figure 4: Concept groups that emerge in different layers in a trained Alexnet. The size of the circle represents the relevance of a concept group in a layer.

field of a filter  $f$ , its activation maps is computed over a set of images. The maximum activation (max-activation) value for each activation map over all the pixels is extracted. The top  $K$  images are then selected corresponding to the activation maps with  $K$  largest max-activation values. For each image, regions with “high” unit activations are identified using a threshold  $T_f$ —taken from [Bau *et al.*, 2017]. This activated region is then scaled up to the image resolution and the filter  $f$ ’s receptive field can be visualized over the selected images. Overall, this qualitative analysis gives a better understanding of a filter by focusing on important areas in the image [Zhou *et al.*, 2014]. We compare the visualizations of the filters of a CNN trained with weight decay against a CNN trained with our proposed losses in Figure 5.

### 4.3 Accuracy Versus Interpretability

Most often, attempts to increase interpretability reduce the discrimination power of a CNN, thereby compromising the validation accuracy of the trained network. We show that by applying soft constraints as proposed in this paper, the validation accuracy is only slightly reduced (less than 1 percent in each case). Results in Table 2 show that most of the unique detectors emerge on the last two convolutional layers of every network. Therefore we compare the ratio of unique detectors (RUD), the number of the unique detectors divided by the total number of filters in a layer, for the final two convolutional layers along with the accuracy of the trained network in Table 3.

## 5 Conclusion

Our formulated losses and regularizer exploit the hidden structures of the data via inducing a more prominent group structure among the filters of CNN. Even simply replacing

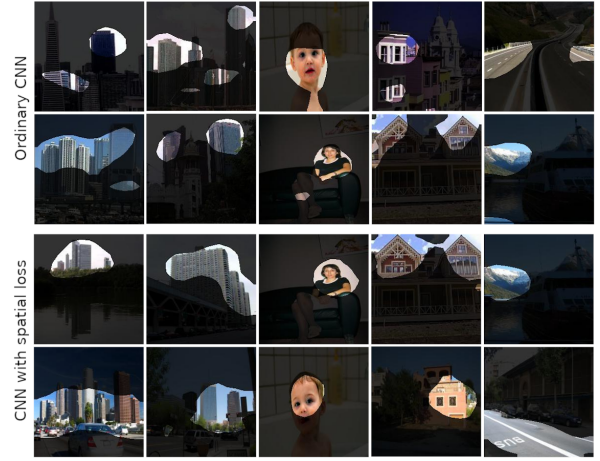


Figure 5: The receptive fields of the filters in the last layer of Alexnet trained with weight decay regularizer (in the top two rows) on Places365 dataset compared with the model trained with spatial loss  $L_s$  (in the bottom two rows).

Dataset	Network	Top-1 acc.	Top-5 acc.	RUD
Places365	Alexnet	51.14%	81.59%	0.176
	Alexnet, ( $R_{bn}$ $L_g$ $L_s$ )	50.46%	80.60%	0.244
	Alexnet-B*	51.32%	81.60%	0.199
	Alexnet-B, ( $R_{bn}$ $L_g$ $L_s$ )	50.46%	80.62%	0.223
	VGG16	54.91%	85.02%	0.189
	VGG16, ( $R_{bn}$ $L_g$ $L_s$ )	54.91%	85.20%	0.210
Imagenet	Alexnet	56.30%	79.04%	0.150
	Alexnet, ( $R_{bn}$ $L_g$ $L_s$ )	55.09%	77.93%	0.164
	VGG16	71.79%	90.45%	0.139
	VGG16, ( $R_{bn}$ $L_g$ $L_s$ )	70.55%	89.87%	0.142

Table 3: A comparison of the Validation Accuracy and RUD Score on training with different regularizers. The RUD score is the ratio of the total number of unique detectors divided by the number of filters in the last two convolutional layers of a CNN.

the conventional weight decay regularizer by our block norm  $R_{bn}$ , we can increase the interpretability of a network without compromising with its discrimination power. We efficiently compute the activation region of a filter over an image in the forward pass as described in Section 3.1, and then constrain the activation regions so that filters in the same group activate similar areas. This further induces a group structure in all the layers of a CNN and enhances interpretability.

## Acknowledgements

The authors gratefully acknowledge support by the Carl-Zeiss Foundation, by the German Research Foundation (DFG) award KL 2698/2-1, and by the Federal Ministry of Science and Education (BMBF) awards 01IS18051A and 031B0770E.

## References

- [Bau *et al.*, 2017] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Computer Vision and Pattern Recognition*, 2017.
- [Deng *et al.*, 2009] 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*, pages 248–255. Ieee, 2009.
- [Huang *et al.*, 2019] Yifeng Huang, Zhirong Tang, Dan Chen, Kaixiong Su, and Chengbin Chen. Batching soft iou for training semantic segmentation networks. *IEEE Signal Processing Letters*, 27:66–70, 2019.
- [Ioffe and Szegedy, 2015] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML’15, pages 448–456. JMLR.org, 2015.
- [Krizhevsky *et al.*, 2017] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.
- [Li *et al.*, 2017] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets, 2017.
- [Mahendran and Vedaldi, 2016] Aravindh Mahendran and Andrea Vedaldi. Visualizing deep convolutional neural networks using natural pre-images. *International Journal of Computer Vision*, 120(3):233–255, 2016.
- [Mocanu *et al.*, 2018] Decebal Constantin Mocanu, Elena Mocanu, Peter Stone, Phuong H Nguyen, Madeleine Gibescu, and Antonio Liotta. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. *Nature communications*, 9(1):1–12, 2018.
- [Tychsen-Smith and Petersson, 2018] Lachlan Tychsen-Smith and Lars Petersson. Improving object localization with fitness nms and bounded iou loss. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [Wen *et al.*, 2016] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. *Advances in neural information processing systems*, 29:2074–2082, 2016.
- [Yosinski *et al.*, 2014] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3320–3328. Curran Associates, Inc., 2014.
- [Zhang *et al.*, 2017] Quanshi Zhang, Wenguan Wang, and Song-Chun Zhu. Examining CNN representations with respect to dataset bias. *CoRR*, abs/1710.10577, 2017.
- [Zhang *et al.*, 2018] Quanshi Zhang, Ying Nian Wu, and Song-Chun Zhu. Interpretable convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [Zhang *et al.*, 2020] Quanshi Zhang, Xin Wang, Ying Nian Wu, Huilin Zhou, and Song-Chun Zhu. Interpretable cnns for object classification, 2020.
- [Zhou *et al.*, 2014] Bolei Zhou, Aditya Khosla, Àgata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. *CoRR*, abs/1412.6856, 2014.
- [Zhou *et al.*, 2017] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.
- [Zhou *et al.*, 2018] Bolei Zhou, David Bau, Aude Oliva, and Antonio Torralba. Interpreting deep visual representations via network dissection. *IEEE transactions on pattern analysis and machine intelligence*, 2018.