Explaining Distributed Neural Activations via Unsupervised Learning

Soheil Kolouri
HRL Laboratories, LLC
3011 Malibu Canyon Rd
Malibu, CA, 90265
skolouri@hrl.com

Charles E. Martin
HRL Laboratories, LLC
3011 Malibu Canyon Rd
Malibu, CA, 90265
cemartin@hrl.com

Heiko Hoffmann
HRL Laboratories, LLC
3011 Malibu Canyon Rd
Malibu, CA, 90265
hhoffmann@hrl.com

Abstract

Recent work has demonstrated the emergence of semantic object-part detectors in activation patterns of convolutional neural networks (CNNs), but did not account for the distributed multi-layer neural activations in such networks. In this work, we propose a novel method to extract distributed patterns of activations from a CNN and show that such patterns correspond to high-level visual attributes. We propose an unsupervised learning module that sits above a pre-trained CNN and learns distributed activation patterns of the network. We utilize elastic non-negative matrix factorization to analyze the responses of a pretrained CNN to an input image and extract salient image regions. The corresponding patterns of neural activations for the extracted salient regions are then clustered via unsupervised deep embedding for clustering (DEC) framework. We demonstrate that these distributed activations contain high-level image features that could be explicitly used for image classification.

1. Introduction

Convolutional Neural Networks (CNNs) provide state-of-the-art performance in a wide variety of computer vision applications including image classification [20, 10], object detection [8, 19], action recognition [23], and segmentation [16]. Deeper, wider, and more complex CNNs are solving vast number of challenging problems in an end to end manner. Despite the undeniable success of CNNs, explaining their inner workings of such networks remains a challenge.

Recent attempts on understanding/explaining the internal representation of the CNNs are mainly focused on inverting and visualizing the mined information in CNNs' features [17, 4, 27, 3]. Perturbing the input and analyzing the network activations is another interesting approach which has been used in similar applications [1, 18]. Our objective in this paper is to go beyond activation analysis of single neurons, and study the patterns of activations in a deep CNN.

Our approach identifies regions of the input image that a trained CNN deems as important or salient. We denote these sub regions of the input image as '*visual attributes*'. In other words, visual attributes are objects/parts that are implicitly represented in activation patterns of a pre-trained CNN. Zhou et al. [28], for instance, showed that a CNN trained for scene classification implicitly learns features for object detection (e.g. beds, couches, etc) and Gonzalez-Garcia et al [9] showed that CNNs indirectly learn semantic parts. The core idea behind our proposed method is to organize these implicit attributes in an unsupervised fashion in order to be able to explain the meaning of these distributed activations. The term 'distributed activations' denotes the multilayer neuronal activations in a deep network.

Our proposed method consists of four main phases. In phase one a top-down approach is utilized to pinpoint the salient parts of the input image based on the network activation patterns. In short, we use elastic non-negative matrix factorization (NMF) together with an of-the-shelf blob detector to obtain multi-scale salient regions of the input image. The NMF components of activation provide localized, tightly clustered, and blob-like regions that correspond to different semantic attributes in the input image. In phase two, a bottom-up approach probes the CNN and extracts hierarchical responses of the network to individual visual attributes. More precisely, for salient patches of different size, we perform global average pooling (GAP) [10, 29] at different convolutional layers of the network and extract fixed size multi-layer features. Note that the extracted GAP features represent various patterns of activations in the network. In the third phase, an iterative unsupervised learning approach is applied to the features extracted from all visual attributes to identify the core activation patterns learned by the network. In the final phase, the input image is summarized by a feature indicating the presence of various core patterns of activations in the network. This feature is an explicit representation for various distributed activation patterns that are present in the network and we denote it as the

'bag-of-neural-activations' features.

The rest of the paper is structured as follows. In the next section, we review related work on understanding the internal representation of trained CNNs. In Section 3, we formulate the problem and provide detailed explanation of the proposed method. Our experimental results are included in Section 4. Finally, we conclude our work in Section 5.

2. Related work

A large body of recent work has focused on understanding the internal representation of trained CNNs. Zhou et al. [28] studied the activation patterns of a CNN trained for scene classification and showed that various object detectors emerge from the activation patterns. In other words, the activation patterns in the CNN encode features for various objects and can be used to detect them; even though the network was not specifically trained for an object detection task.

In [17, 4], the authors propose methods for inverting CNN feature representations to obtain images which contain the information preserved by the network. In another approach, Zeiler & Fergus [27] introduced a method to visualize image patterns that activate each hidden unit in a CNN. Furthermore, Generative networks [5, 3] have also attracted much attention recently as they could be used to visualize and understand convolutional networks.

Other approaches alter the input images (e.g., by blocking a small portion of the image) and study the differences between corresponding CNN outputs to decode semantic value of the alternation [18, 1]. Bazzan et al. [1] for instance, use a pre-trained CNN and mask-out parts of the input image and analyze the recognition score (i.e. network response); in this manner they are able to localize the object in the input image. Alternatively, Zhou et al. [29] used global average pooling to model convolutional activation patterns and localized objects recognized by the network in this manner.

Our proposed method is more related to the methods proposed by Gonzalez-Garcia et al [9], and Zhou et al [28]. Zhou et al. [28] extracted regions of an input image which correspond to highest activations in different units of a CNN and asked workers on Amazon Mechanical Turk (AMT) to identify the common theme or concept that exist between these top scoring regions. In a similar approach, Gonzales-Garcia et al [9] asked human annotators whether the image regions with highest activations cover the same visual concepts. The mentioned methods, utilize corresponding image regions with highest activations of each unit of a CNN. In contrast to the mentioned methods, we propose to model the pattern of activations in a group of units as opposed to single units. Our major contributions in this paper are:

1. Proposing an approach for obtaining salient regions of an image using the activation patterns of a pre-trained deep CNN via NMF

2. Unsupervised learning of distributed patterns of neural activations in a CNN

3. Proposed method

We provide an unsupervised scheme for identifying the learned key distributed neural activations of a Convolutional Neural Network (CNN). We start by identifying the regions of the input image that are deemed salient by the network, and then analyze the network's activation patterns in these salient regions. Our method consists of four steps: 1) extraction of salient regions of the input image, 2) representing distributed neural activations as multi-level GAP features, 3) unsupervised clustering of distributed activation patterns, and 4) producing a bag-of-neural-activations feature that complements the CNN features. Figure 1 illustrates the steps involved in mining the key distributed activations from a pretrained CNN. Figure 1 (a) demonstrates the extraction of salient attributes, while 1 (b) illustrates the unsupervised extraction of the key activation patterns. The convolutional activations shown in Figure 1 (a) are the output of the pretrained network at the last convolutional layer and before the last max/average-pool. Each step is described below.

3.1. Salient attribute extraction

We first identify salient regions of an input image. Various publications have recently addressed the problem of utilizing CNNs for saliency prediction [12, 15]. These methods often train a CNN to focus (i.e. larger activations) on regions of an image that are considered salient by humans (i.e. humans fixate on those regions). Our goal in this Section, however, is different in the sense that we are interested in finding the salient parts of an input image as perceived by the pre-trained CNN, not by humans. This step is crucial in our framework as it parses the input image into small regions containing salient parts from which the distributed neural activations will be extracted.

The over arching idea here is to decompose the convolutional responses of the final layer of a CNN, and represent it as a linear combination of several components. If these components are sparse and spatially localized, then they can be used to parse various regions of the input image. To achieve this goal we use elastic Nonnegative Matrix Factorization (NMF) [2, 7], which is shown to provide components containing localized, tightly clustered, and blob-like regions [21, 11]. In short, given a pretrained CNN and an input image, we apply elastic NMF [2, 7] to the activation patterns (i.e. last convolutional layer) of the CNN to obtain and extract principal activation patterns for the input data.

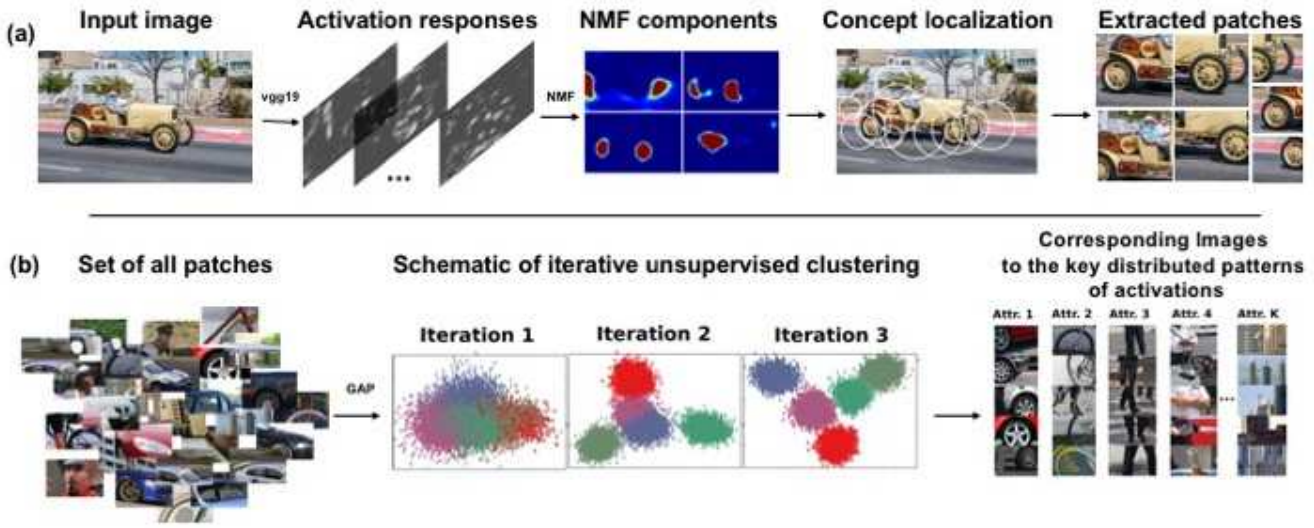

Figure 1: Schematic of the unsupervised learning of distributed patterns of activations in a deep CNN. Given a deep CNN (in this case VGG19 [20]) we first extract image parts that the CNN deems salient (a), then an unsupervised method is utilized to identify the key patterns of distributed activations (b). Image patches are resized for visualization purposes.

Note that since we do not use the fully connected layers of the CNN at this stage, the size of the input image could vary.

More precisely, let $X = [x_k]_{k=1}^{m} \in \mathbb{R}^{d \times m}$ denote the vectorized CNN responses of the last convolutional layer (e.g. the 'conv5_4' of VGG19), where m is the number of convolutional kernels at the last layer (e.g. $m = 512$ in VGG19), and d is the number of nodes per convolutional kernel and scales with the size of the input image. Then the NMF is formulated as,

$$\begin{aligned} \text{argmin}^*_{W,H} \quad & \frac{1}{2}\|X - HW\|_F^2 + \\ & \gamma(\lambda)(\|W\|_1 + \|H\|_1) + \\ & \frac{1}{2}\gamma(1-\lambda)(\|W\|_F^2 + \|H\|_F^2) \\ s.t. \quad & W \geq 0,\ H \geq 0 \end{aligned} \tag{1}$$

where $\|.\|_F$ is the Frobenius norm, $\|.\|_1$ is the elementwise L_1 norm, columns of $H \in \mathbb{R}^{d \times r}$ are the non-negative components, $W \in \mathbb{R}^{r \times m}$ is the non-negative coefficient matrix, r is the rank of matrix H, which corresponds to the number of extracted components, and λ and γ are regularization parameters. We use a coordinate descent solver to find H and W.

Note that the objective function in Eq. 1 does not have an explicit regularizer to enforce spatial smoothness of components, however, it has been shown that NMF components often exhibit blob-like high spatial connectedness [21, 11]. We use this characteristic of the extracted components, H, to identify salient regions of the input image. After upsampling each component (i.e., resizing to the original image size to counter the effect of pooling layers) we process it by a Laplacian-of-Gaussian blob-detector [14] to extract regions of the input image that are considered salient by CNN. Figure 2 summarizes above process, and shows the NMF components and the extracted blobs for an input image. Notice the spatial connectedness and blob-like nature of the NMF components. Finally, we should point out that the extracted patches are at different scales and their sizes vary.

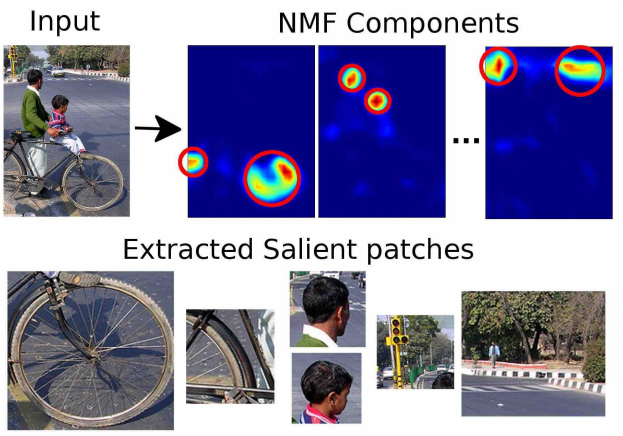

Figure 2: The input image to CNN, the NMF components applied to the last convolutional layer, detected blobs, and the corresponding salient patches are demonstrated.

3.2. Distributed neural activations via GAP features

We would like to obtain a representation for the distributed activations of the CNN for each extracted patch.

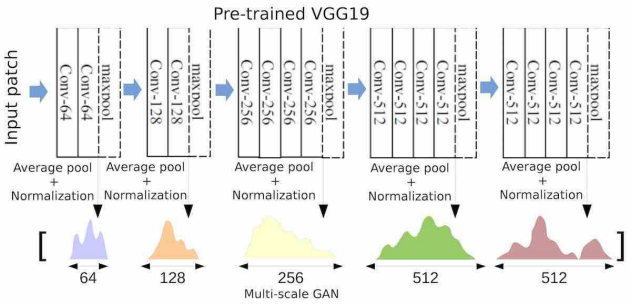

Figure 3: Extracting the multi-scale global average pooling (GAP) feature from image patches.

The challenge here is two fold: 1) obtain the responses of the CNN for the various size input patches, and 2) find a compact representation for these activations. Note that resizing the patches is not a feasible solution as it would introduce scale variability and unnecessary smoothness due to interpolation. To address these issues we follow the ideas presented in fully-convolutional neural networks [13] and GoogLeNet [22] and use global average pooling (GAP) over different layers of the CNN. Our approach here can be considered as an extension of the GAP-CNN introduced by Zhou et al. [29].

Having in mind that our goal is not to train a new CNN but to explain the distributed activations of a pre-trained CNN, we probe the activation patterns of the CNN at different layers and construct a multi-scale feature for the extracted patches. More specifically, we perform average pooling at each layer of the network right before the 'max pooling', normalize the GAP features, and concatenate them to obtain a representation for distributed activations of the network (See Figure 3). The proposed feature captures the response energy of various convolutional kernels at different layers, and provides a succinct representation of the CNN. The normalization is needed so the scale of average poolings at different layers are the same (i.e. range is zero to one). Figure 3 illustrates the extraction of the proposed multi-scale GAP feature from an input image patch using VGG19 [20].

3.3. Unsupervised clustering of salient attributes

Given the GAP features, which represent distributed neural activations of the CNN, from all extracted salient patches, we utilize an unsupervised learning framework to cluster these activation patterns. There exist many options for unsupervised clustering including k-means, spectral clustering, sparse and elastic subspace clustering techniques [26, 6], and deep unsupervised clustering techniques [25, 24]. In this work, we use the unsupervised deep embedding for clustering (DEC) [24] to cluster the GAP features.

The idea behind DEC [24] is to transform the data into a linear/nonlinear embedding space with richer data representation and cluster the data in that space. The embedding and the clusters are then learned simultaneously in an iterative fashion. More precisely, let the GAP features extracted from salient patches be $\{z_i \in Z\}_{i=1}^{N}$, instead of clustering in Z, DEC transfers the data to a lower-dimensional space,V, via $f_\theta : Z \rightarrow V$ and finds cluster centroids $\{\mu_j \in V\}_{j=1}^{K}$ in V. Here, we define the mapping, f_θ, to the embedding space in its most general form as a parametric function with parameters θ; f_θ can be chosen to be as simple as a linear mapping or as complex as a deep neural network.

Given an initial estimation of the mapping f_θ (i.e. random initialization of θ) and the initial cluster centroids $\mu_j|_{j=1}^{K}$, let $Q \in \mathbb{R}^{N\times K}$ with elements $q_{i,j}$, denote the soft assignment of the GAP+CNN features to the cluster centroids. The student t-distribution or Gaussian distribution could be used as a kernel to define such a soft assignment. We followed the work of Xie et al. [24] and utilized the student t-distribution :

$$q_{i,j} = \frac{(1 + \frac{\|v_i - \mu_j\|^2}{\rho})^{-\frac{\rho+1}{2}}}{\sum_{k=1}^{K}(1 + \frac{\|v_i - \mu_k\|^2}{\rho})^{-\frac{\rho+1}{2}}} \tag{2}$$

where ρ is the degree of freedom of the distribution, and $v_i = f_\theta(z_i)$. Ideally one would like to obtain clusters with high confidence (i.e. tight clusters with well-separated centroids). Let $P \in \mathbb{R}^{N\times K}$ be the unknown ideal soft data assignment matrix. Then one can define a loss function for learning the function parameters, θ as a dissimilarity measure between P and Q, for instance the Kullback-Leibler (KL) divergence:

$$KL(P|Q) = \sum_{i=1}^{N}\sum_{j=1}^{K} p_{i,j} log(\frac{p_{i,j}}{q_{i,j}}) \tag{3}$$

or similarly the cross entropy function. Note that Q depends on the mapping parameters θ. Hence if the ideal soft data assignment matrix, P, was known then the function parameters could have been optimized with respect to the above loss function. Xie et al. [24] propose to estimate P from Q at each iteration using,

$$p_{i,j} = \frac{q_{i,j}^2 / f_j}{\sum_{k=1}^{K} q_{i,k}^2 / f_k} \tag{4}$$

where $f_j = \sum_{i=1}^{N} q_{i,j}$. Note that P pushes probabilities toward 1 and 0, and also prefers balanced clusters (clusters with same number of members). Finally, having an estimation for P at each iteration we use stochastic gradient descent (SGD) to minimize Eq. 3 with respect to the parameters embedding parameters θ. This will lead to the iterative unsupervised clustering scheme presented in Algorithm 1.

Algorithm 1 Deep Embedding for Clustering

1: Initialize θ
2: **repeat**
3: $v_i = f_\theta(z_i)$, for $i = 1, ..., N$
4: Calculate μ_j, for $j = 1, ..., K$ using K-means
5: Calculate Q from Eq. 2
6: Estimate P from Q using Eq. 4
7: $\theta \leftarrow \theta - \epsilon \frac{\partial KL(P|Q)}{\partial \theta}$
8: **until** Achieving a local minima

The outcome of the unsupervised deep embedding method is a mapping, f_θ, that embeds the input GAP features into a discriminant subspace, together with the key distributed activations (i.e. cluster centers), μ_j for $j = 1, ..., k$.

3.4. Bag of neural activations

In the final phase of our proposed method, we obtain a feature that represents the presence of various key distributed activations in an image. For a given input image, we identify its salient regions with elastic NMF, extract its GAP features from the M identified salient regions v_i for $i = 1, ..., M$ (M could vary for different input images), map the features to the embedding space via f_θ, and obtain their cluster membership. Using the cluster memberships, we generate the histogram of key distributed neural activations presented in an image. We denote this K-dimensional feature vector as '*bag-of-neural-activations*'.

Finally, to measure the importance of the distributed neural activations in the network, we perform classification solely based on the bag-of-neural-activations that explicitly represent the existence of various distributed neural activation patterns. Moreover, to understand the amount of information overlap between the bag-of-neural-activations and the deep CNN features (i.e. the last fully connected layer) we concatenated these features and performed classification based on the concatenated feature. The end to end system is shown in Figure 4. A test input image goes through the CNN and our proposed system. The final classification accuracy is shown for the CNN, the bag-of-neural-activations, and the concatenated features obtained from both systems.

4. Experimental Results

In our experiments we used two image datasets. The first dataset is a subset of the ImageNet dataset and is solely used for a proof of concept of our method. The dataset contains 6 classes related to autonomous driving, namely, 'car', 'construction', 'pothole', 'bicycle', 'traffic-cop', and 'crowd' with an average 1000 images per class. For the second dataset, we used the Stanford car dataset, which contains 16,185 images of 196 classes of cars, where classes are typically at the level of Make, Model, and Year (e.g., 2012 Tesla Model S).

Half of the data was used for training and the rest for testing. For both datasets, we fine-tuned VGG19 networks [20], which were pre-trained on the ImageNet dataset. The trained networks served as the baseline for our experiments.

Given the trained CNNs, we follow the steps as shown in Figure 4. First, the convolutional activations are fed to an elastic NMF. An NMF component is a weighted sum of activation responses that satisfies sparseness, spatial coherency, and captures salient variations in the activation patterns. In our experiments we observed that the NMF components consistently contained semantically meaningful regions of the input image. The number of NMF components was chosen based on the average reconstruction error of activation patterns of the training set. We discovered that seven components for the first dataset (i.e. part of ImageNet), and five components for the second dataset (Stanford car dataset) satisfied the reconstruction error criteria (i.e., 95% of variation). Given that the NMF components demonstrate a blob-like spatial consistency, we used an off-the-shelf Laplacian-of-Gaussian blob detector [14] on NMF components to extract salient regions (patches) of the input image.

For each extracted image region the GAP responses at different layers of the CNN are extracted. The GAP features enable us to obtain a fixed size feature vector despite variability in the size of the salient regions (See Figure 3). Using VGG19 and extracting normalized GAP responses at different layers right before the max pool leads to a feature vector of size 1472 (i.e., $64+128+256+512+512$). Each element of the GAP feature indicates the average activation response of the input image with respect to one of the convolutional kernels, and therefore it could be considered as a feature for distributed neural activations.

Next, we cluster the GAP features from all salient regions extracted from training images. To do so we utilized DEC. For the mapping to the embedding space, f_θ, we utilized a linear function $f_\theta(v_i) = Wv_i$. The dimension of the embedding was cross-validated over the following set $d \in \{2^n : n = 1, ..., 8\}$, we found out that $d = 32$ for both datasets led to the lowest Fisher score. It should be mentioned that, for the choice of the mapping to the embedding space, f_θ, in our experiments we also tested a neural network of depth three, but we observed no negligible improvement over the outcome of the linear alternative.

After clustering the GAP features (i.e. the distributed neural activations) we visualized the corresponding images to the top 5 cluster members (i.e., closest to the cluster center as measured by Q in Equation 2) of 21 sample classes extracted from the first dataset. It can be seen that these distributed neural activations correspond to semantically similar image regions, even though their visual appearance dif-

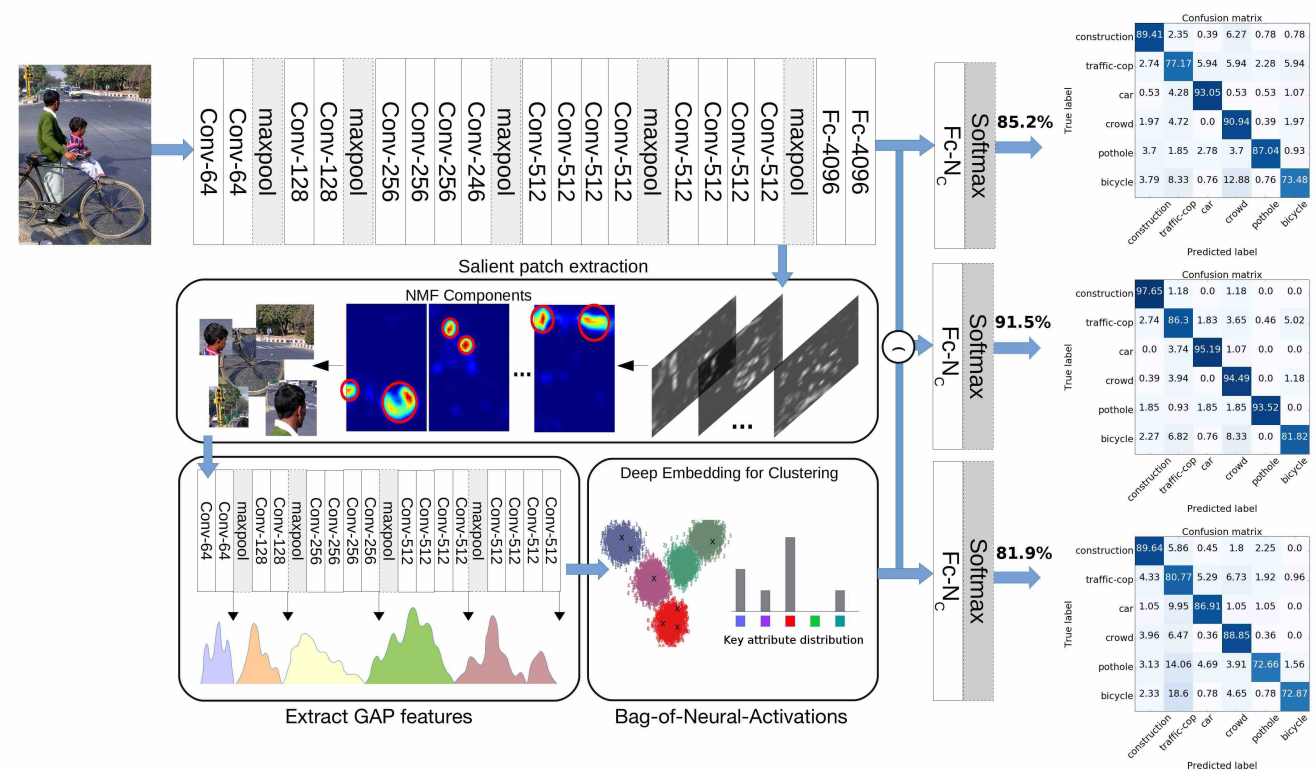

Figure 4: Supplementing the VGG19 network with the bag-of-neural-activations features will lead to in average a 42.5% reduction in classification error. The '(' symbol represents concatenation.

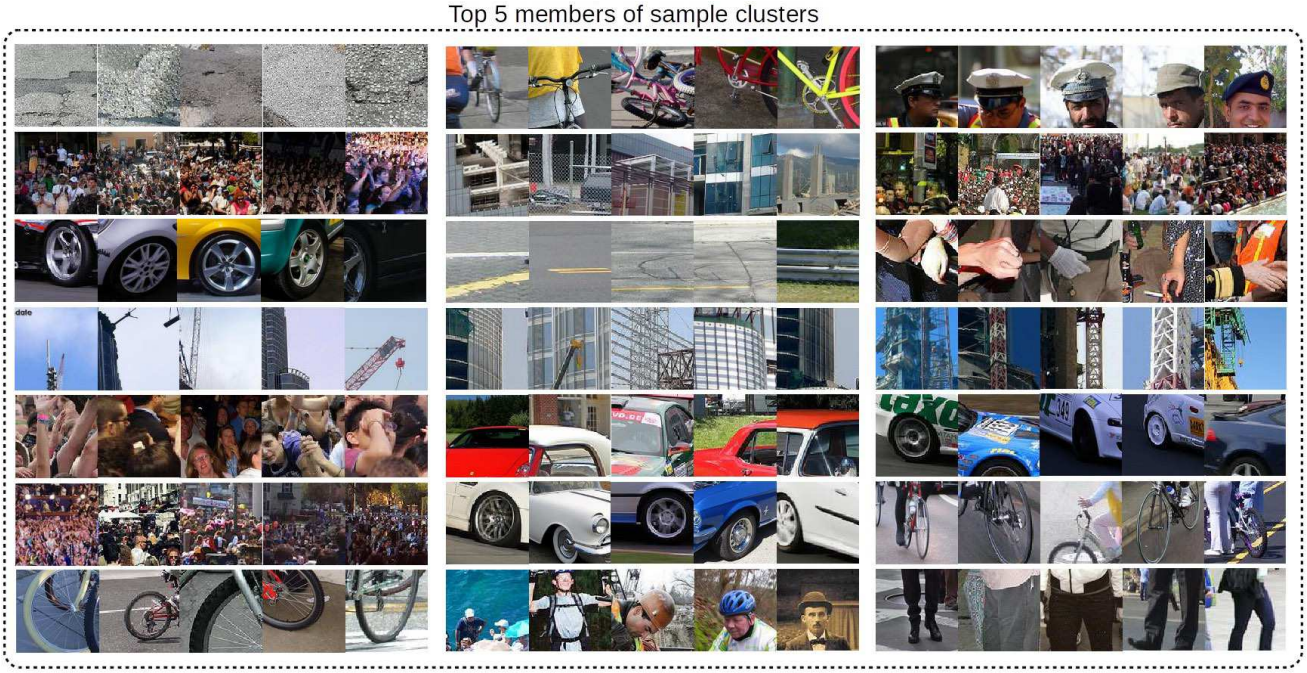

Figure 5: Corresponding images for the top 5 members of sample distributed activation clusters calculated from VGG19 on part of the ImageNet dataset. **The patches are resized to a fix size for the purpose of visualization.**

fers significantly. Figure 5 shows that the distributed activation patterns in the network implicitly encode visual attributes such as: car wheel, bike wheel, road, construction crane, legs, hands, and hats while it has not been specifically trained to recognize these visual attributes. These attributes were not labelled (e.g., with bounding-boxes or pixel-wise segmentation) in any of the input images used in our experiments. Figure 6 shows the corresponding images for the top 5 cluster members for the Stanford car dataset. It can be seen that our approach decodes a wide variety of fine-grained attributes, such as headlight shapes, grille shapes, and vehicle emblems that are implicitly encoded in the CNN.

Finally, the bag-of-neural-activations features are concatenated to the features extracted from the CNN right before the softmax classifier. The softmax layer is then retrained based in the new concatenated feature. Figure 4 shows the classification accuracy and confusion matrices based on the VGG19 feature, the bag-of-neural-activations feature, and the VGG19+ bag-of-neural-activations feature for the first dataset. For Dataset 1 (ImageNet), combining the bag-of-neural-activations features with the VGG19 features led to a 42.5% reduction in classification error. Note that we compared the SOA on the same reduced data set and therefore the comparison is fair.

Furthermore, we repeated the same experiment with various CNNs. Specifically, we used Residual Networks [10] of different depth, namely ResNet50, ResNet101, and ResNet152, which were pretrained on the ImageNet dataset. We then repeated the classification experiment by complementing the features extracted from the network by the bag-of-neural-activation features and observed consistent improvement for all networks. The results are reported in Table 1. We note that Top-N denotes the accuracy when the true label is in the top N predictions. The evaluation server for the Stanford car dataset only provided Top 1 classification accuracy and hence the Top 2 classification accuracies are not provided for this dataset.

For the Stanford car dataset we retrained a VGG19 to classify the 196 classes on the training set. The network was pretrained on ImageNet. We cropped the images to the provided bounding box and simply resized the images to 224×224. The classification accuracy of the trained VGG19 and the VGG19+bag-of-visual-features are also reported in Table 1. The top-two classification results are not reported as the evaluation server for the dataset only reported the top 1 classification accuracy. On this dataset, combining the bag-of-neural-activations with the VGG19 features led to a 28.6% reduction in classification error, which highlights one of the benefits analyzing the distributed neural activations of a CNN.

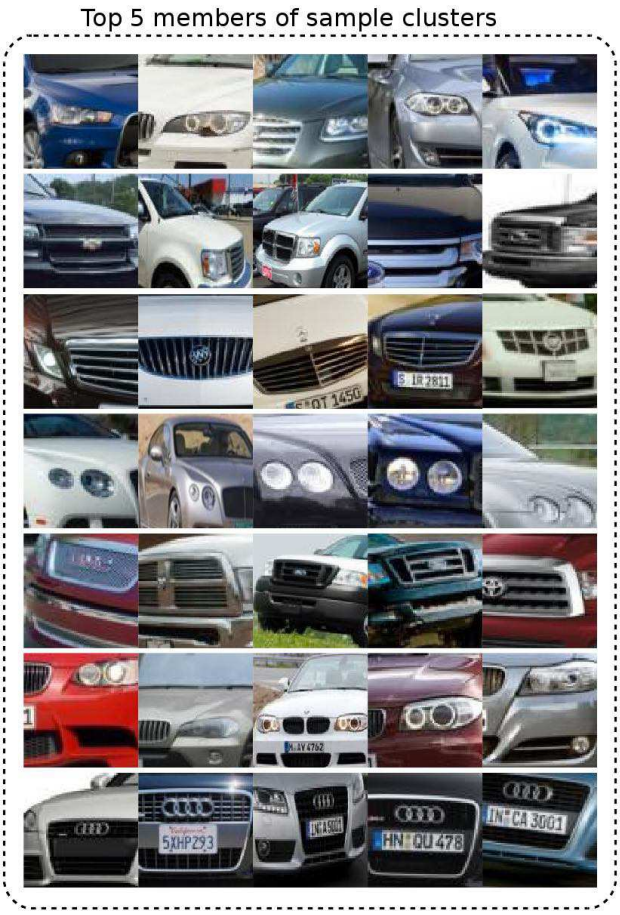

Figure 6: Corresponding images for the top 5 members of sample distributed activation clusters calculated from VGG19 on the Stanford car dataset. **Note that the patches are resized to a fix size for the purpose of visualization.**

5. Discussion

In this paper, we introduced an unsupervised method for clustering distributed neural activations in a CNN. We showed that these distributed activations in the network correspond to visually understandable concepts. Moreover, we demonstrated that an explicit utilization of these implicit patterns of activations leads to a boost in classification accuracy on two datasets. Existing work in the literature has focused on extracting salient attributes within images by identifying image regions corresponding to the largest activations of a single convolutional kernel within a CNN [9, 28]. In contrast, our approach models distributed activation patterns of a CNN to learn key visual attributes implicitly presented in the network and enhances the robustness of the CNN through these attributes.

We demonstrated that utilizing an unsupervised learning method on top of a pretrained CNN allows one to learn ac-

Classification accuracy

	Dataset 1		Dataset 2
	Top 1	Top 2	Top 1
BONA	81.9%	93.5%	57.23%
VGG19	85.18 %	93.42%	62.47%
VGG19+BONA	91.49%	97.23%	73.21%
ResNet50	91.11%	96.8%	-
ResNet50+BONA	92.00%	97.75%	-
ResNet101	91.96%	96.45%	-
ResNet101+BONA	92.8%	97.66%	-
ResNet152	92.02%	97.14%	-
ResNet152+BONA	92.85%	97.75%	-

Table 1: Classification accuracy based on our bag-of-neural-activations (BONA) features calculated from VGG19, based on features extracted from various CNNs, and based on the combination of BONA and CNNs' features for the two datasets used in our experiments.

tivation patterns of the network which leads to mining the high-level visual attributes learned by a CNN. We visualized the corresponding images for top members of each distributed activation cluster within an embodiment of VGG19 that had been pretrained on the ImageNet and the Stanford car datasets. We demonstrated that, despite the vast amount of visual variability in the datasets, the extracted clusters of distributed neural activations correspond to semantically meaningful image regions.

Finally, we proposed a new feature based on the multi-layer distribution of activation patterns, denoted as 'bag-of-neural-activations', and showed that complementing a CNN with the bag-of-neural-activations feature leads to higher classification accuracies for both datasets and for different CNN architectures. The idea of learning distributed activation patterns within pre-trained CNNs and linking the patterns to semantically meaningful attributes opens the door to more explainable machine learning systems.

References

[1] L. Bazzani, A. Bergamo, D. Anguelov, and L. Torresani. Self-taught object localization with deep networks. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9. IEEE, 2016. 1, 2

[2] A. Cichocki and P. Anh-Huy. Fast local algorithms for large scale nonnegative matrix and tensor factorizations. *IEICE transactions on fundamentals of electronics, communications and computer sciences*, 92(3):708–721, 2009. 2

[3] E. L. Denton, S. Chintala, R. Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in neural information processing systems*, pages 1486–1494, 2015. 1, 2

[4] A. Dosovitskiy and T. Brox. Inverting visual representations with convolutional networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 1, 2

[5] A. Dosovitskiy, J. Tobias Springenberg, and T. Brox. Learning to generate chairs with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1538–1546, 2015. 2

[6] E. Elhamifar and R. Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE transactions on pattern analysis and machine intelligence*, 35(11):2765–2781, 2013. 4

[7] N. Gillis and S. A. Vavasis. Fast and robust recursive algorithmsfor separable nonnegative matrix factorization. *IEEE transactions on pattern analysis and machine intelligence*, 36(4):698–714, 2014. 2

[8] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015. 1

[9] A. Gonzalez-Garcia, D. Modolo, and V. Ferrari. Do semantic parts emerge in convolutional neural networks? *arXiv preprint arXiv:1607.03738*, 2016. 1, 2, 7

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015. 1, 7

[11] P. O. Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of machine learning research*, 5(Nov):1457–1469, 2004. 2, 3

[12] S. S. Kruthiventi, V. Gudisa, J. H. Dholakiya, and R. Venkatesh Babu. Saliency unified: A deep architecture for simultaneous eye fixation prediction and salient object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5781–5790, 2016. 2

[13] M. Lin, Q. Chen, and S. Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013. 4

[14] T. Lindeberg. *Scale-space theory in computer vision*, volume 256. Springer Science & Business Media, 2013. 3, 5

[15] N. Liu, J. Han, D. Zhang, S. Wen, and T. Liu. Predicting eye fixations using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 362–370, 2015. 2

[16] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015. 1

[17] A. Mahendran and A. Vedaldi. Understanding deep image representations by inverting them. In *2015 IEEE conference on computer vision and pattern recognition (CVPR)*, pages 5188–5196. IEEE, 2015. 1, 2

[18] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. *arXiv preprint arXiv:1604.07379*, 2016. 1, 2

[19] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 1

[20] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1, 3, 4, 5

[21] A. Sotiras, S. M. Resnick, and C. Davatzikos. Finding imaging patterns of structural covariance via non-negative matrix factorization. *NeuroImage*, 108:1–16, 2015. 2, 3

[22] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015. 4

[23] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 4489–4497. IEEE, 2015. 1

[24] J. Xie, R. Girshick, and A. Farhadi. Unsupervised deep embedding for clustering analysis. *arXiv preprint arXiv:1511.06335*, 2015. 4

[25] J. Yang, D. Parikh, and D. Batra. Joint unsupervised learning of deep representations and image clusters. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 4

[26] C. You, C.-G. Li, D. P. Robinson, and R. Vidal. Oracle based active set algorithm for scalable elastic net subspace clustering. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 4

[27] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pages 818–833. Springer, 2014. 1, 2

[28] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Object detectors emerge in deep scene cnns. *arXiv preprint arXiv:1412.6856*, 2014. 1, 2, 7

[29] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 1, 2, 4