Understanding CNN via Deep Features Analysis

Hao Xu, Yueru Chen, Ruiyuan Lin and C.-C. Jay Kuo
University of Southern California, USA
E-mail: iamxuhao@gmail.com, {yueruche, ruiyuanl}@usc.edu, cckuo@sipi.usc.edu

***Abstract*—Two quantitative metrics are proposed for evaluating trained deep features at different convolution layers in this work. We first show mathematically that the Gaussian confusion measure (GCM) can be used to identify the discriminative ability of an individual feature. Next, we generalize this idea, introduce another measure called the cluster purity measure (CPM), and use it to analyze the discriminative ability of multiple features jointly. The discriminative ability of the trained Convolution Neural Network (CNN) features is confirmed by experiments. Further studies utilizing GCM and CPM as tools offer important insights into the CNN, such as understanding the behavior of trained CNN features and the good detection performance of some object classes that were considered difficult in the past. Finally, the trained deep feature representation is compared between different CNN structures to validate the superiority of deeper networks.**

I. Introduction

The Convolution Neural Network (CNN) performs strongly in the detection of objects due to the discriminant power of its superior features, which are automatically obtained from a large amount of training data. Great efforts have been exerted to explain CNN's performance theoretically and experimentally. The need of nonlinear activation was explained in [16], [17]. In order to analyze deep features, visualization methods [21], [32], [35] reconstruct the input images or visualize the responses to provide insights into the mechanism behind the activation of a certain filter. Despite their success in explaining CNN's dominating performance, such analysis is less scalable given the increasing number of object classes and deeper network structures. In [1], the average precision (AP) of an individual filter is computed and leveraged as a quantitative measurement of a trained CNN filter's discriminative ability. An important contribution of their work is to confirm the existence of "GMC-like features".

Human brain cells that only respond to specific and complex visual stimuli (such as the face of one's grandmother) are called the grandmother cells (GMC) [22]. They have been studied in artificial neural networks and are believed to play a critical role in object recognition. For example, the cat filter was extensively studied in [19] for its resemblance to the cat-face GMC. The term "GMC-like features" is used to refer to deep features or feature groups that respond solely to one object class. A similar concept is adopted in the standard feature encoding for image classification [13], [33].

In Fig. 1, two conv_5 filters from the CaffeNet are used to demonstrate the importance of studying the GMC-like features. From left to right, the filters correspond to the "dining table" and "dog" object class, respectively. In the left, no dining table image is among the top nine activations and there is a significant overlap in the Gaussian confusion plot between the filter responses of the dining table and non-dining table classes. In the right, all top nine activations are dog images and there is little overlap in the Gaussian confusion plot between the filter responses of the dog and non-dog classes. The visual inspection of the top nine activations in Fig. 1 demonstrates the superior discriminative ability of GMC-like features, but such inspection cannot be conducted in large scale networks due to the infeasible workload on the human inspector. Note that the Gaussian confusion plot reflects the important message of the top nine activations and, therefore, motivates us to develop the Gaussian confusion measure (GCM, see Sec. III-A), which yields a score representing the discriminative ability of a deep feature without human intervention.

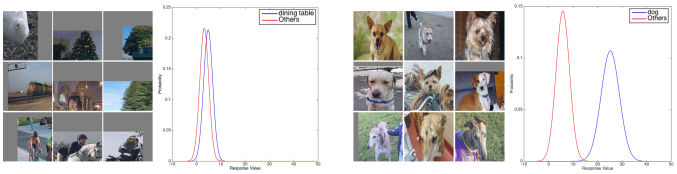

Fig. 1. None-GMC-like and GMC-like features are compared in the left and right subfigures. The first and third subfigures give the top nine activations for two conv_5 filters and the second and fourth subfigures show their Gaussian confusion plot, which reflects the distribution of a filter's response value (see discussion in Sec.III-A). In this example, the GMC filter (i.e., filter 142 in conv_5 of the fast-RCNN [9] CaffeNet model) is mainly activated by the dog class and its Gaussian confusion plot shows better separation.

For certain object classes, one deep feature is not a GMC-like feature by itself (see Sec. IV-B). However, when a group of deep features work together, they behave as a GMC-like feature group. To understand the observation, we develop another measure called the cluster purity measure (CPM, see Sec.III-B) to gauge the group performance of deep features. As opposed to training an SVM for a subset of features and evaluating the resulting average precision, CPM can be directly computed and not subject to stochastic difference.

Generally speaking, our research goal is to gain a better understanding of the CNN by evaluating its trained deep features and studying the GMC-like deep features. It has three major contributions. First, we propose the Gaussian confusion measure (GCM) to automatically evaluate the performance of an individual filter. Second, we propose the cluster purity measure (CPM) to automatically evaluate the group performance of a set of filters. They are both simple and efficient to train, and are useful in automatic evaluation of the deep features.

Third, we study the CaffeNet and the VGG_M_1024 architectures by evaluating its filters using the proposed metrics. Our study provides an explanation of their stellar performance, reveals CNN's capability in detecting certain object (e.g. the dining table, see Sec.IV-B), identifies potential issues with deep feature representations, and verifies the benefits in using deeper networks.

The rest of this paper is organized as follows. Related work is reviewed in Sec. II. Two metrics are introduced in Sec. III to evaluate whether a deep filter or a group of deep filters qualifies as the GMC-like feature. Experimental results based on the evaluation of the GMC-like features are reported in Sec. IV. Concluding remarks are provided in Sec. V.

II. Related Work

To better understand CNN deep features and their representations, [26], [34], [35] presented methods to find locally optimal visual inputs for individual filters. Despite starting from different input priors, these methods were capable of selecting the image which can maximally activate a filter. By visually examining the visualization results, one can verify the existence of GMC-like features as well as identify the targeted image content associated with the GMC-like feature. Besides visually inspecting features, Agrawal, et al. [1] evaluated conv_5 features with respect to their Average Precision (AP) on the PASCAL-VOC 2012 dataset, and showed the existence of GMC-like features for several object classes.

Since Girshick *et al.* [10] introduced the RCNN to solve the object detection problem, many efforts have been done to improve its efficiency and accuracy. The spatial pyramid pooling (SPP) was proposed in [12] to share convolution computations. Girshick [9] added the bounding box regression and extended the finetuning to all layers for better accuracy. The time-consuming region proposal extraction was replaced by the region proposal network to speed up detection to near real-time in [23]. Deeper network configurations [27], [29] offered even better performance in the benchmarking datasets at the price of a higher training cost with more parameters. Although the performance can be further improved with larger networks, we are more interested in understanding the reason of their success. By understanding the strength and weakness of a network, it helps stretch the performance furthermore at a reasonable cost of added parameters.

The exact purpose of the filters in a convolution layer has been studied since the early days of CNN research. It was observed in [10], [18] that the conv_1 features are low level features such as edge, color, or texture. The deeper convolution layers extract higher level features by assembling lower level features in the previous layer with different weights [20]. The pooling layers provide spatial and possibly rotation invariance to feature extraction [3]. The features assembly was studied in [28]. It was concluded that only a local subnetwork is activated or trained for a specific input pattern. The selection is done through local competition of several activation functions such as rectified linear, maxout and local winner-take-all. Research in [11] reveals the CNN model structure by formulating the CNN as a deformable part model, where the convolution layers serve as part-feature detectors and the pooling layer uses a distance transform to assign different weight to part-detectors according to their relative locations. An interesting study conducted in [36] indicates that object detectors could be trained to classify scene images, which have certain similarity as part-detectors are trained to recognize objects.

III. Evaluation Metrics

To train a CNN for object detection, each training sample contains an object in a region of interest (ROI) with class label c. The filter responses in convolution layers indicate deep features. In conv_1, filters act as low-level feature extractors. As more information is aggregated in deeper convolution layers, filters become more discriminative and class-specific. As a result, filters will be activated by more specific inputs.

The problem of interest can be stated as follows. Given a ROI which has an object inside, we extract one value from the filter response at the conv_5 layer by max pooling. There are 256 filters at the conv_5 layer and, consequently, we obtain a response vector of dimension 256 for each input ROI. Our question is whether there exists one or a group of filters that can help separate an object class from other classes. In the following, we propose two metrics that offer a quantitative answer to this question.

A. Gaussian Confusion Measure (GCM)

We use N testing data samples to test a CNN for a specific object class, say, class c. The testing samples consist of both positive ones that are in class c and negative ones that are not in class c. To analyze a specific filter, we take its response value from the 256-D response vector and collect N response values, each is associated with a class label. They can be classified into two groups according to their class labels; namely, in class c or not in class c. Consequently, we can draw two distribution curves of response values for the two groups. This explains the Gaussian confusion plots in Fig. 1, where the y-axis is the normalized histogram value and the x-axis is a specific filter response value. As shown in Fig. 1, each filter response can be well approximated by a Gaussian (or normal) function denoted by $\mathbb{N}(m, \sigma)$, where m is the mean and σ is the standard deviation.

We can state the problem mathematically below. For a set of testing samples $(\vec{x}_i, y_i)$, where $\vec{x}_i$ is the ROI and y_i is the class label, we use $\mathbb{F}_i(f_{kj})$ to denote the response value of f_{kj} which is the jth filter in the kth convolution layer. We group the response values according to their class labels into $\mathbb{C}$ groups, where $\mathbb{C}$ is the number of class labels in the testing dataset.

To get the GCM from a given filter, f_{kj}, and a given object class, $c \in \mathbb{C}$, we need to compute the mean and standard deviation of two hypotheses: H_0 and H_1, where H_0 states that the object belongs to class c while H_1 states that the object does not belong to class c. M_0 and M_1 indicate the number of objects belonging to the two hypotheses, respectively.

The mean and standard deviation of hypothesis H_0 can be computed as

$$m_0(f_{kj}, c) = \frac{1}{N_0} \sum_{\forall i, s.t. y_i = c} \mathbb{F}_i(f_{kj}), \quad (1)$$

$$\sigma_0(f_{kj}, c) = \sqrt{\frac{1}{N_0} \sum_{\forall i, s.t. y_i = c} [\mathbb{F}_i(f_{kj}) - m_0(f_{kj}, c)]^2}. \quad (2)$$

The mean and standard deviation of hypothesis H_1 can be computed as

$$m_1(f_{kj}, c) = \frac{1}{N_1} \sum_{\forall i, s.t. y_i \neq c} \mathbb{F}_i(f_{kj}), \quad (3)$$

$$\sigma_1(f_{kj}, c) = \sqrt{\frac{1}{N_1} \sum_{\forall i, s.t. y_i \neq c} [\mathbb{F}_i(f_{kj}) - m_1(f_{kj}, c)]^2}. \quad (4)$$

To decide whether a filter is activated, we should choose a threshold denoted by t. Then, the false positive is the probability of the Gaussian distribution $\mathbb{N}(m_1, \sigma_1)$ falls in $[t, +\infty)$. The false negative is the probability of the Gaussian distribution $\mathbb{N}(m_0, \sigma_0)$ fall in $(-\infty, t]$. The sum of these two error types can be written as:

$$\begin{aligned} \mathbb{Q}_{kj}(t, c) = &\int_{-\infty}^{t} \mathbb{N}(m_0(f_{kj}, c), \sigma_0(f_{kj}, c)) \\ &+ \int_{t}^{\infty} \mathbb{N}(m_1(f_{kj}, c), \sigma_1(f_{kj}, c)), \end{aligned} \quad (5)$$

It is typical to set

$$t = \frac{1}{2}[m_0(f_{kj}, c) + m_1(f_{kj}, c)]. \quad (6)$$

The specific object class that minimizes $\mathbb{Q}_{kj}(t, c)$ in Eq.(5) is a candidate class for us to search for its GMC response. Mathematically, we have

$$c_{gmc} = \arg\min_{c \in \mathbb{C}} \mathbb{Q}(t, c). \quad (7)$$

Finally, we define the Gaussian confusion measure (GCM) of filter f_{kj} with respect to its corresponding c_{gmc} as:

$$\mathbb{GCM}(f_{kj}) = \frac{\mathbb{Q}_{kj}(t, c_{gmc})}{\mathbb{E}_j\{\mathbb{Q}_{kj}(t, c_{gmc})\}} \Big/ \frac{m_0(f_{kj}, c_{gmc})}{\mathbb{E}_j\{m_0(f_{kj}, c_{gmc})\}}. \quad (8)$$

The numerator and the denominator of GCM in Eq.(8) are the normalized decision error and the normalized mean of the null hypothesis, respectively. Intuitively, a GMC-like feature should have a higher mean response, $m_0(f_{kj}, c)$, due to higher correlation between objects of the same class. Besides, it should have a lower decision error $\mathbb{Q}_{kj}(t, c)$ since it can differentiate positive and negative samples at higher accuracy. As a result, GCM approaches 0 and a larger number for GMC-like and non-GMC-like features, respectively. This is supported by experimental results given in Fig. 2, where we show both the mean response value and the decision error of 256 conv_5 features in a fast-RCNN trained CaffeNet with 40,000 iterations. Only three filters match the above description and the GCM successfully picks them out.

The GCM provides an effective feature evaluation tool based on the statistics of the response values of a filter given different inputs. Its validity is verified via applying Gaussian tests to the response values extracted from Pascal[6] dataset. It is also more thorough than evaluating the filter by APs, since the GCM can model the underlining statistics of the filter (rather than testing the filter with a dataset). However, GCM is limited to the examination of a single filter. Thus, a new metric is needed for evaluating the group discriminative ability of a number of filters.

B. Cluster Purity Measurement (CPM)

The Cluster Purity Measurement (CPM) is proposed to evaluate how multiple filters perform jointly in differentiating object classes. As compared with the AP evaluation method introduced in [1], the CPM metric does not require the separate training of an SVM.

To evaluate the CPM score on a set of n filters $f_{k\vec{j},\ \vec{j} \in \mathbb{J}}$ in the kth convolution layer ($\mathbb{J}$ represents the set of filter indexes), the testing samples $(\vec{x_i}, y_i)$ are organized into two groups, Ω_c and $\Omega_{\not c}$. The first group contains testing samples belonging to class c while the second group contains testing samples not belonging to class c. In testing, the filter responses vector, $\vec{\mathbb{F}}_i(f_{k\vec{j}})$, corresponding to the selected set of filters can be obtained at the conv_k layer. The responses vectors are n-dimensional vector. They can be split into two groups according to if the testing sample came from Ω_c or $\Omega_{\not c}$. An example of the response vector in the $2D$ case is shown in Fig. 3.

In operation, the CNN prefers a high response from a filter when it "fires" on an input. The recent success in the Binary network[4] further proves the fact that most of CNN filter's behavior can be considered as a ON/OFF switch. This motivates the CPM metric for a set of filters that better discriminates a certain object class jointly. For this reason, we define a characteristic point at $P = \max_i\{\vec{\mathbb{F}}_i(f_{k\vec{j}})\}$ for an object class, where the max operation is taken over each dimension in the filter responses vector in the n-dimensional space.

As shown in Fig. 3, most points closer to P correspond to the "car" testing samples. To quantitatively measure this property, we define the CPM metric as a measure of the percentage of points corresponding to the target object class within the top K closest points to point P. Mathematically, we have

$$\mathbb{CPM}(\vec{\mathbb{F}}(f_{k\vec{j}}), c) = \frac{N_c}{K}, \quad (9)$$

where N_c is the number of responses vectors $\vec{\mathbb{F}}_i(f_{k\vec{j}})$, which are within the top K points closest to P, and correspond to testing samples with class label $y_i = c$. We set $K = 200$ empirically in the experiment.

In practice, we use the GCM metric to rank features' discriminative ability on each object class individually and, then, use the CPM metric to evaluate the joint discriminative ability of a selected group of deep features that perform well on a certain object class.

IV. Experimental Results

We use the latest fast-RCNN [9], [14] implementation as the baseline. Since our objective is to analyze deep features, we

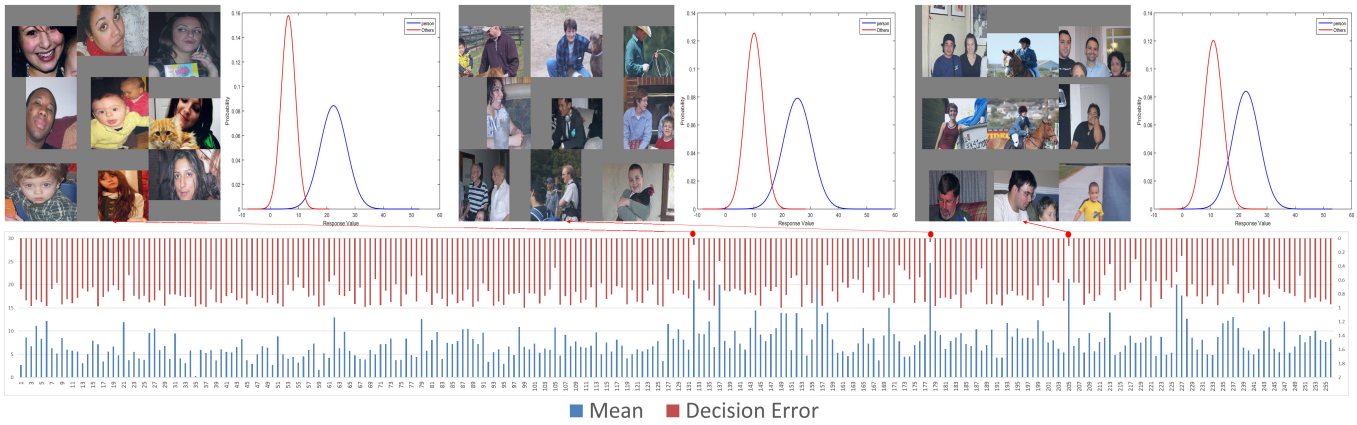

Fig. 2. The red and blue bars correspond to the decision error, $\mathbb{Q}_{kj}(t,c)$, and the null mean response, $m_0(f_{kj},c)$, of the 256 conv$_5$ filters of the fast-RCNN CaffeNet with 40,000 iterations, respectively. The blue bar is plotted bottom-up using the main vertical axis while the red bar is plotted top-down using the secondary vertical axis. The red dots indicate filters of indices 132, 178, and 205 (from left to right), which have the GCM score of 0.05, 0.9, and 0.17, respectively. These filter responses have low decision errors so that they are selected as GMC-like features. The top 9 activations and the Gaussian confusion plot for each of these three cases are presented for validation.

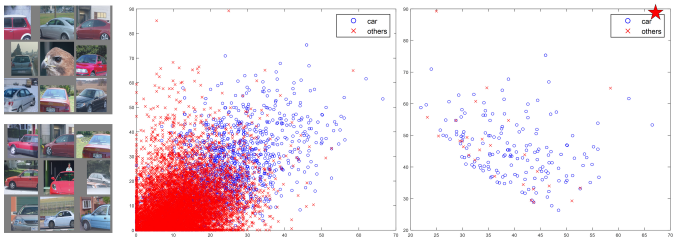

Fig. 3. The response vectors of filter 188 (top left) and 212 (bottom left) in the 40000 iteration CaffeNet model are plotted in the $2D$ space. The blue dots in the plot corresponds to response vectors obtained from testing samples of cars and the red dots corresponds response vectors obtained from others. The plot in the right takes the top 200 responses vectors that are closest to the point P (shown as the star on the top right). The CPM score in this example is 0.87.

disable the bounding box regression to avoid confusion. The analyzed network structure is the CaffeNet and VGG_M_1024, pretrained with the ImageNet CLS dataset [25]. To visualize filter responses (i.e., deep features) and top activations, we use the deconv network [35] and visualization toolbox from [34]. Furthermore, deconv images and top 9 activations are extracted separately using the PASCAL-VOC 2012 dataset with the whole image as the ROI. To save space, we also generate the 20 classes Gaussian confusion plot by combining all 20 individual Gaussian confusion plots.

A. *Evaluating GMC Features*

Our experiment begins with applying the two developed metrics, GCM and CPM, to the conv$_5$ features extracted from the CaffeNet. The fast-RCNN does not crop out ROIs until it reaches the fully connected layer. Thus, we created a new testing configuration, where the network ends at the ROI-pooling layer to obtain network responses at a given ROI. The ROI-pooling layer will output one maximum response value for each filter, which is used to evaluate whether the filter is activated or not. That is, for all 256 filters in the conv$_5$ layer, we have a 256-D feature vector to denote the response of each of the filters.

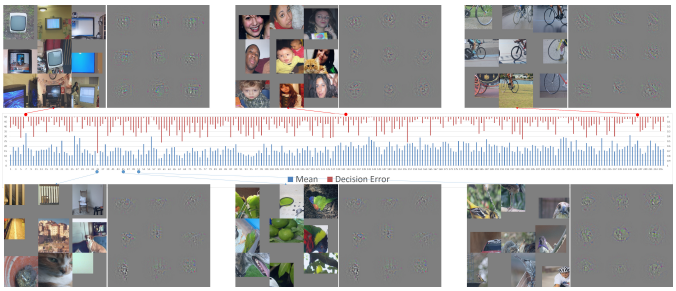

Fig. 4. The red and blue bars correspond to the decision error, $\min_c \mathbb{Q}_{kj}(t,c)$, and the null mean response value, $m_0(f_{kj},c)$, of the 256 conv$_5$ filters of the fast-RCNN CaffeNet with 40,000 iterations, respectively. (Note that these bars are different from those in Fig. 2, where $\mathbb{Q}_{kj}(t,c)$ and $m_0(f_{kj},c)$ were plotted for $c =$"person"). The red bar is plotted top-down using the secondary vertical axis while the blue bar is plotted bottom-up using the main vertical axis. The red dots highlight filters 7, 132, and 246 from left to right. These filters have low decision errors and are selected as GMC-like features. The blue dots highlight filters 35, 45, and 51. These filters have higher decision errors and their responses are selected as none-GMC-like features. The top 9 activations and their deconv results are presented to validate whether it is a GMC-like or none-GMC-like feature.

A few examples of GMC and none-GMC like features are demonstrated in Fig. 4, where the top 9 activations of the GMC-like features are all of the same object classes, and their GCM values indicate that these filters do not confuse other object classes. The activations of none-GMC-like features vary on the other hand. From left to right, filter 35 seems to look for the vertical line structure, filter 45 seems to look for the green color, filter 51 seems to look for defocused background. Due to space limit in the paper, the full filter GCM results are included in the supplemental material as the appendix.

B. Evaluating GMC Feature Groups

Besides individual GMC-like features, the proposed CPM can find some interesting GMC-like feature groups. The top nine activations and the 20 classes Gaussian confusion plot of filter 188, 212, and 172 in conv_5 are shown in Fig. 5.

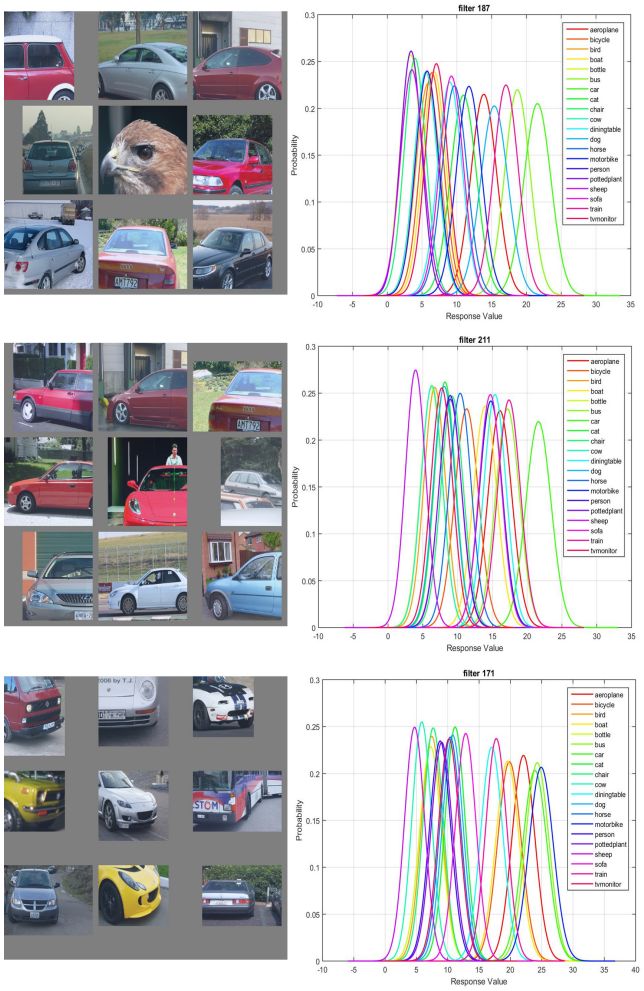

Fig. 5. From left to right: the top 9 activations and the 20 classes Gaussian confusion plot of filters 188, 212, and 172 in conv_5 of the fast-RCNN CaffeNet with 40,000 iterations. The CPM score is 0.54 for filter 188 alone, 0.86 for filter 188 and filter 212 - it reaches 0.90 for the three filters group. The filters correspond to the tail, window, and head of the car, respectively.

The three filters are the top contributors of a deep feature group for the "car" object. Individually, each of them has a high decision error. They can get confused with different object classes (e.g., "bird" and "bus"). However, when the three work together, the discriminative power improves a lot and the deep feature group will mostly respond to the "car" images. As compared with an individual GMC, another advantage of the deep feature group is its robustness against occlusion. The GMC-like features are extremely sensitive to occlusion. This is a real issue in general object detection since objects are often partially occluded in images. Since the GMC-like feature group can respond to a subset of object's components, its detection performance will be more robust when occlusion occurs.

C. Evaluating Context Features

Detection of a "dining table" and a "chair" is interesting as there is no clear definition of the object. Before applying the CNN to object detection, the average precisions of using the DPM [7] in detecting a dining table and a chair are both low. They are 14.7% and 17.2%, respectively, in the PASCAL-VOC 2012 dataset [6]. Recently, the fast-RCNN significantly boosts the corresponding performance to 40.7% and 67.9%. We observe from our experiment that the difference in the performance gain is due to "context features" trained by the network.

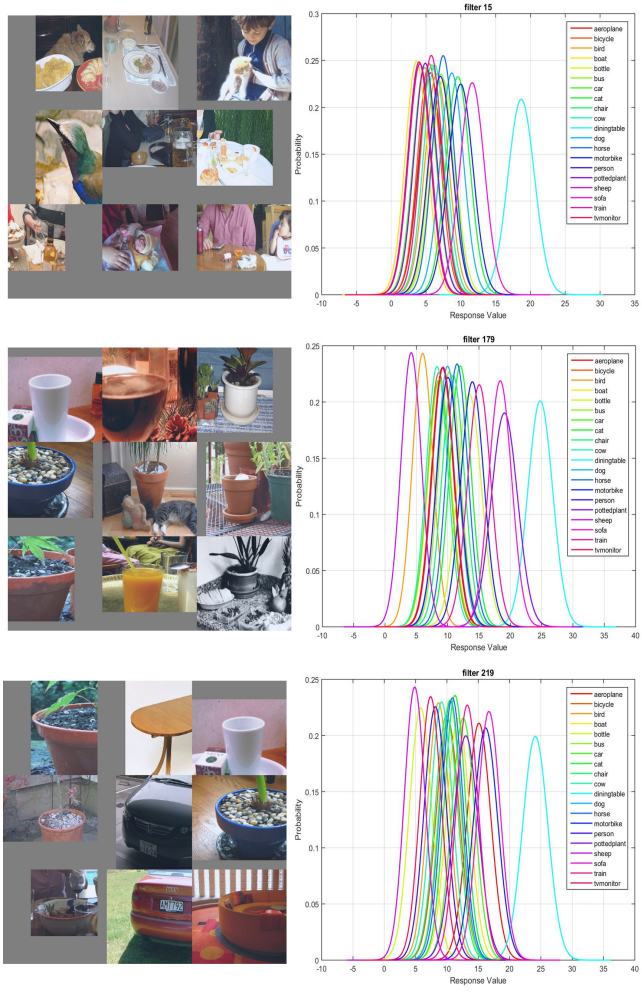

Fig. 6. From left to right: the top 9 activations and the 20 classes Gaussian confusion plot of filters 16, 180, and 220 in conv_5 of the fast-RCNN CaffeNet Model with 40,000 iterations. The CPM score is 0.36. These filters correspond to dishes, cups, an ellipse contour, respectively. Filters 16 and 180 are not related to a dining table itself but rather objects placed on top of it.

By identifying a group of deep features that corresponds to detecting "dining table" and plotting their top nine activations and the 20 classes Gaussian confusion plot in Fig. 6, we see

that the key information used in detecting a dining table is cups and/or dishes placed on top of it. That is, while the DPM detector attempts to model the shape or contour of an object, there is little success in man-made objects due to intra-class variation. The CNN approaches this problem by summarizing the function of the dining table, which is to hold cups or dishes on top of it. This is quite close to how humans define a dining table: "an object to put dishes on".

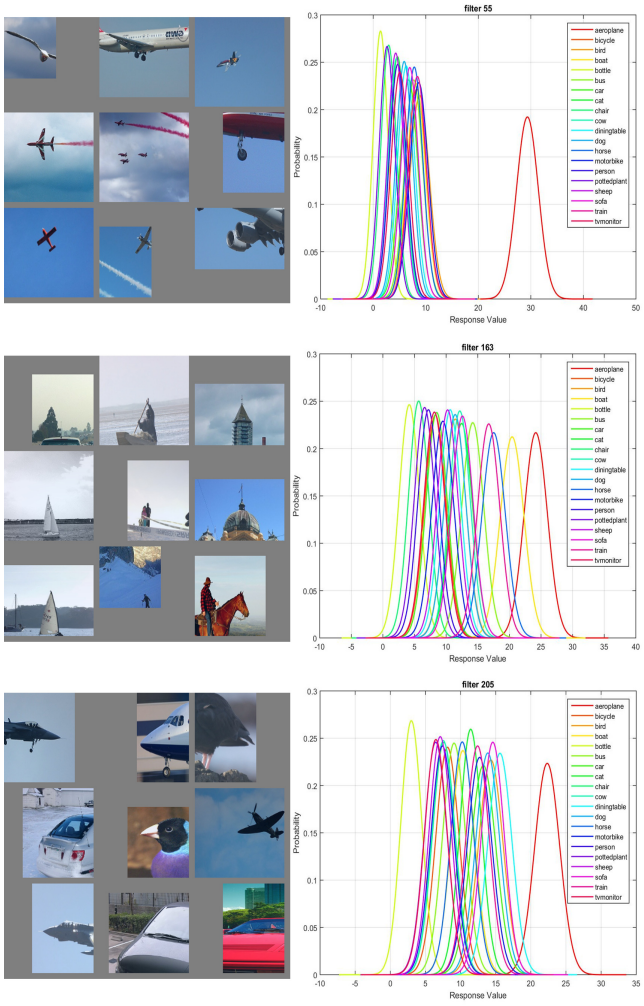

Fig. 7. The top 9 activations and the 20 classes Gaussian confusion plot of filters 56, 164, and 206 in conv_5 of the fast-RCNN CaffeNet Model with 40,000 iterations are shown from left to right. The CPM score is 0.61. These filters are all dedicated to detecting blue or gray color in the input, which are the typical background associated with "airplanes".

Besides context features that correspond to the function of an object, CNN also learns context features that associate with the most probable background of an object. In Fig. 7, the three deep features have a CPM score of 0.61. However, the top nine activations and the 20 classes Gaussian confusion plot both confirm that the deep features are not learned to detect "airplanes" but the background.

D. Evaluating Shared Features

The CNN also learns shared deep features that have superior discriminative ability over a few object classes. In Fig. 8, the top nine activations and the Gaussian confusion plot verify that the underlying feature tries to detect the wheel of a motorbike or a bicycle. Although the top nine activations include examples of the wheel of cars, the Gaussian confusion plot indicates that the filter is less capable in differentiating cars from others. As shown, the deep feature can differentiate motorbikes and bicycles from other objects, but cannot separate them well. This example demonstrates the first type of shared deep features, which correspond to certain shared characteristics across different object classes such as a common part of different objects.

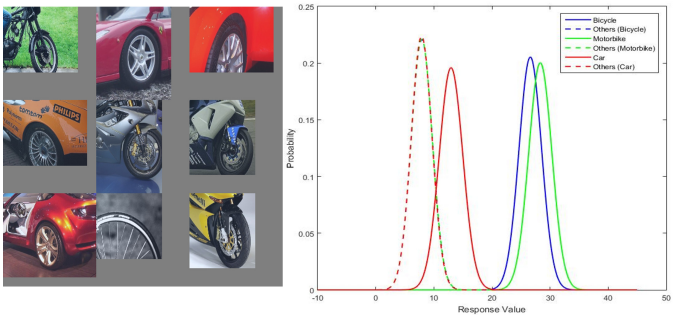

Fig. 8. From left to right: 1)the top nine activations of filter 216 in conv_5 layer of the 40000 iteration CaffeNet model, and 2) the Gaussian confusion plot of the bicycle v.s. others, the motorbike v.s. others, and the car v.s. others.

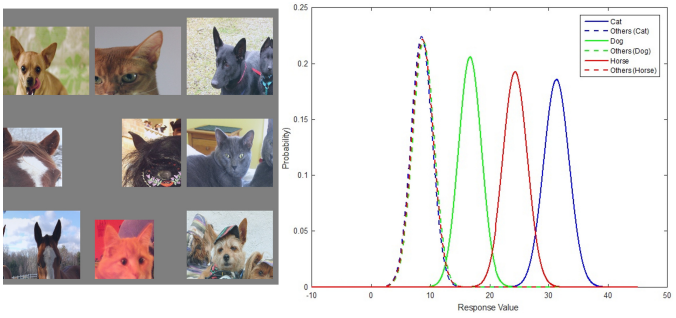

Fig. 9. From left to right: 1) the top nine activations of filter 141 in conv_5 layer of the 40000 iteration CaffeNet model, and 2) the Gaussian confusion plot of the cat v.s. others, the horse v.s. others, and the dog v.s. others.

There is another type of shared deep features, which not only correspond to multiple object classes but have the discriminative ability to separate object classes. In Fig. 9, the top nine activations and the Gaussian confusion plot verify that the feature tries to detect the ear of cats, dogs or horses. Unlike the previous example, this feature has different response value ranges for the three object classes and can separate them with small decision errors. Since this feature's discriminative ability is closely related to the actual response values, it will be deprecated in the binary network [4]. This explains why the

binary network could only achieve similar performance as the floating point network but could not outperform it.

E. Other Deep Features

Besides the deep features that correspond to one or several object classes, the CNN learns other features that seem to have little discriminative ability between classes. Usually, these deep features correspond to generic texture, shape or color. In Fig. 10, the top nine activations indicate that the filter is looking for mesh-like texture. The corresponding Gaussian confusion plot indicates that the feature can hardly differentiate between any classes by itself.

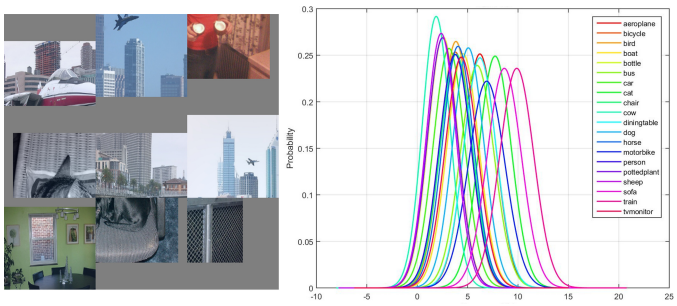

Fig. 10. From left to right: 1)the top nine activations of filter 214 in conv_5 layer of the 40000 iteration CaffeNet model, and 2) the Gaussian confusion plot of all 20 object classes.

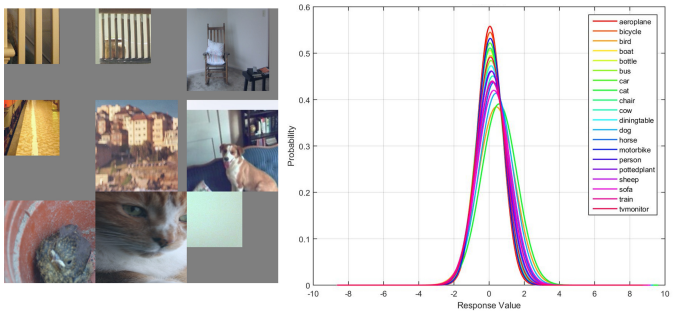

Fig. 11. From left to right: 1)the top nine activations of filter 35 in conv_5 layer of the 40000 iteration CaffeNet model, and 2) the Gaussian confusion plot of all 20 object classes.

There is also another type of filter, which is confirmed to be an "unused" filter for classification. An example is shown in Fig. 11, where the top nine activations indicate that the filter is looking for parallel vertical patterns. However, the Gaussian confusion plot of all 20 object class shows that the deep feature has the same response range for all object classes. By examining the FC_1 weight assigned to this conv_5 feature, we see that it is almost ignored since the weight for this filter is negligible. Although these types of deep features seem to have little use in classification, they contribute strongly for generation tasks. The recent success in style transfer [8] relies heavily on these generic features as they contribute to the computation of the "style loss".

F. Comparison between VGG_M_1024 and CaffeNet

The VGG_M_1024 network structure used in the fast-RCNN benchmark is selected for comparison with the CaffeNet. The two networks start from the same 96 conv_1 features, which are combined into 256 features in conv_2. From that point on, while the CaffeNet has 384, 384, and 256 features in conv_3, conv_4, and conv_5 layers, respectively, the VGG_M_1024 has 512, 512, and 512 features in its corresponding layers. The CaffeNet uses grouping to divide filters into 2 groups due to the GPU memory limit at their development time, which results in divided feature representation that may be less optimal as compared with VGG_M_1024 where all features are combined.

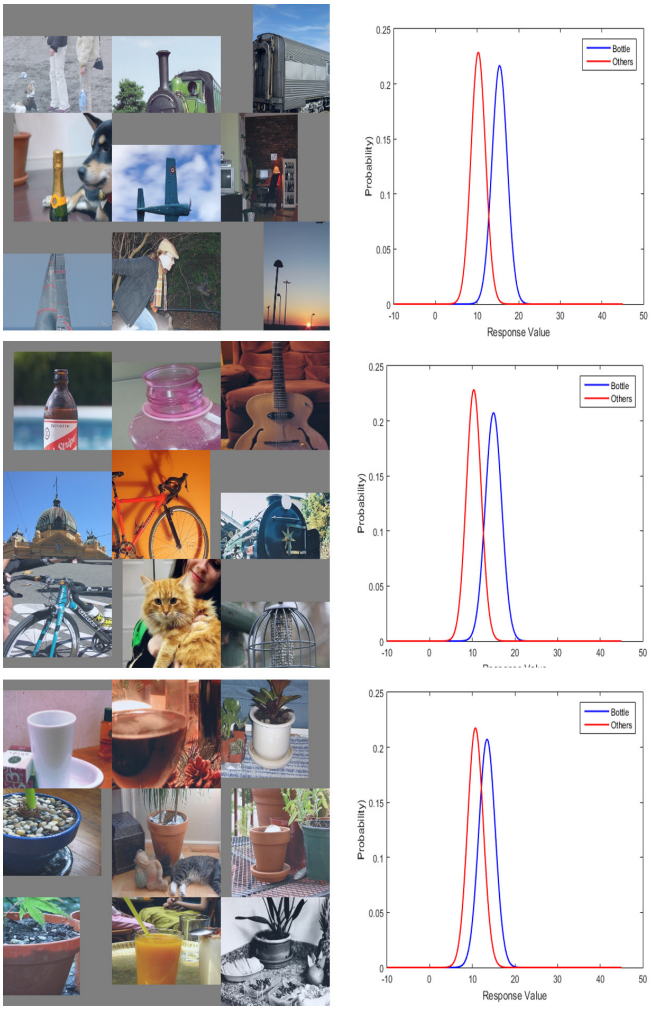

Fig. 12. The top three filters in the CaffeNet trained to detect the "bottle" object. Their corresponding Gaussian confusion plots are shown under the top nine activations.

Due to larger capacity in deeper convolution layers, the VGG_M_1024 model has a higher chance to learn GMC-like features, and the learned deep feature representation is more comprehensive and discriminative. An example is the "bottle" object class, where the CaffeNet has no GMC-like feature while the VGG_M_1024 have two GMC-like features. As

Network	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	motorbike	person	plant	sheep	sofa	train	tv
CaffeNet	0.67	0.54	0.27	0.51	0.21	0.3	0.92	0.70	0.23	0.31	0.33	0.66	0.61	0.46	0.89	0.45	0.51	0.36	0.57	0.63
VGG	0.70	0.60	0.59	0.61	0.37	0.49	0.95	0.86	0.50	0.47	0.55	0.77	0.42	0.49	0.91	0.67	0.53	0.46	0.47	0.68

TABLE I
THE CPM SCORES OF THE TOP 5 FEATURES IN THE CAFFENET AND THE VGG_M_1024 FOR EACH OBJECT CLASS.

	Feature Type	Filter ID	Count
CaffeNet	1 Class GMC	7 16 30 38 53 56 57 59 65 72 74 95 116 130 139 157 158 183 203 205 217 220 233 239 240 241 246 253	28
	Shared and Distinctive	17 26 28 42 43 46 47 86 100 105 128 132 141 142 152 155 161 168 171 178 180 188 189 196 204 208 211 212 213 222 232 236 242 243 251 254	36
	Shared and Not Distinctive	19 55 64 70 83 85 89 101 103 104 117 125 159 160 163 164 166 172 178 187 191 192 215 216 231 244 245 249	28
	Other	the rest of the filters	164
VGG_M_1024	1 Class GMC	28 30 35 44 51 53 77 80 82 84 86 88 92 100 153 160 178 179 190 196 214 233 247 269 288 294 304 341 367 371 390 402 417 427 430 439 446 451 457 461 469 472 473 486 503 510	46
	Shared and Distinctive	36 58 69 74 85 89 91 96 98 105 115 163 164 167 171 193 200 221 228 231 237 243 257 264 265 268 272 290 300 307 331 337 375 376 393 415 422 424 432 444 460 467 471 480 491 496 504 506	48
	Shared and Not Distinctive	2 7 19 29 40 95 99 106 124 132 139 144 145 146 157 185 223 248 260 270 273 276 277 281 287 305 320 325 333 338 342 350 355 369 377 387 420 494 498	39
	Other	the rest of the filters	379

TABLE II
THE FUNCTIONALITY SUMMARY OF CONV$_5$ FILTERS TRAINED BY THE CAFFENET AND THE VGG_M_1024.

shown in Fig. 12 and Fig. 13, the most discriminative features trained by the CaffeNet are not GMC-like features. Thus, the decision error is large. In contrast, there are two GMC-like features trained by the VGG_M_1024, leading to low decision errors. In general, deep features trained from VGG_M_1024 tend to be more discriminative, and more GMC-like features can be found in VGG_M_1024. A summary of the deep feature representation for each object class is concluded in Table I, where the CPM scores of the top 5 deep features for each object class are compared between the CaffeNet and the VGG_M_1024. The comparison further verifies that deeper networks learn a better deep feature representation.

G. Complete Filter Profile

After analyzing the complete set of CNN deep features, we summarize different types of features trained by the CaffeNet and the VGG_M_1024 in TableII, where the "1 class GMC" features correspond to features that differentiate one object class from the others (e.g., the dog example in Fig. 1), the "shared and distinctive" features correspond to features that differentiate several object classes from the others but cannot differentiate within these classes (e.g., example in Fig. 9), the "shared and not distinctive" feature correspond to features that differentiate several object classes from the others and differentiate within these classes (e.g., example in Fig. 8), and the "other" features include the rest of the more generic features.

V. CONCLUSION

Two effective metrics (i.e., GCM and CPM) to evaluate deep features of CNNs were proposed in this work. Thorough studies were conducted on deep features learned from the CaffeNet and VGG_M_1024 network based on these metrics. We identified different deep feature types, compared their discriminative ability, and analyzed the advantages of the deeper network structure in the experiment. Finally, we provided a full conv$_5$ feature profile for both the CaffeNet and VGG_M_1024 to conclude our study.

ACKNOWLEDGMENT

This project was supported by National Heart Lung Institute (R01HL129727)(T.K.H.).

REFERENCES

[1] P. Agrawal, R. Girshick, and J. Malik, "Analyzing the performance of multilayer neural networks for object recognition," in *European Conference on Computer Vision*, 2014.

[2] S. Amari, "Natural gradient works efficiently in learning," *Neural Computation*, 1998.

[3] Y. Bengio, I. J. Goodfellow, and A. Courville, "Deep learning." Book in preparation for MIT Press, 2015.

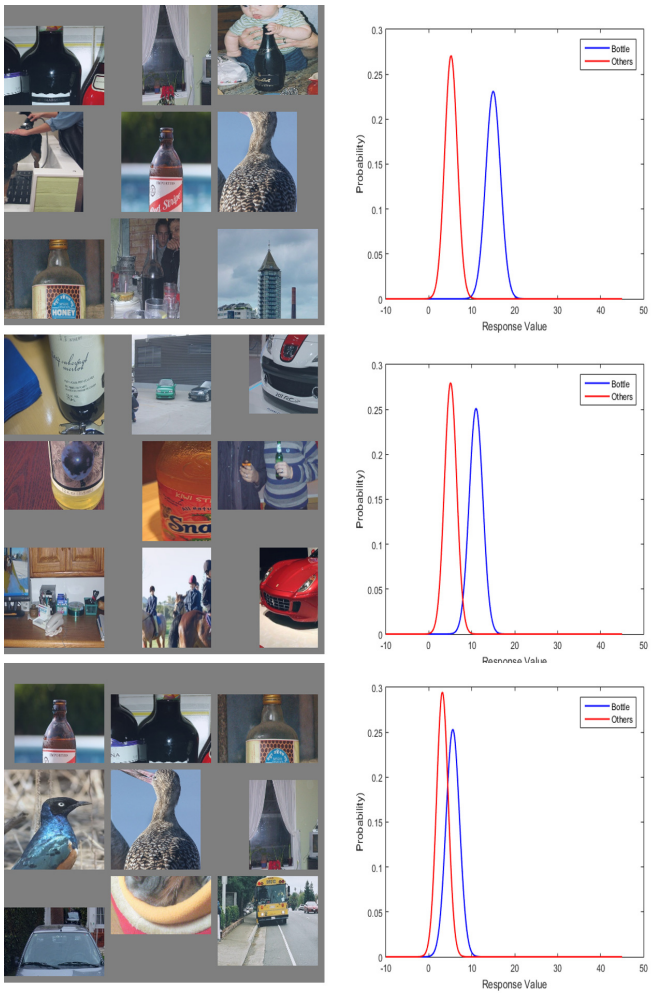

Fig. 13. The top three filters in the VGG_M_1024 (right) trained to detect the "bottle" object. Their corresponding Gaussian confusion plots are shown under the top nine activations.

[4] M. Courbariaux, Y. Bengio, and J.-P. David, "Binaryconnect: Training deep neural networks with binary weights during propagations," in *Neural Information Processing Systems*, 2015.

[5] Y. N. Dauphin, H. de Vries, and Y. Bengio, "Equilibrated adaptive learning rates for non-convex optimization," *arXiv preprint arXiv:1502.04390*, 2015.

[6] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, pp. 303–338, June 2010.

[7] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[8] L. A. Gatys, A. S. Ecker, and M. Bethge, "A neural algorithm of artistic style," *arXiv preprint arXiv:1508.06576*, 2015.

[9] R. Girshick, "Fast r-cnn," *arXiv preprint arXiv:1504.08083*, 2015.

[10] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[11] R. Girshick, F. Iandola, T. Darrell, and J. Malik, "Deformable part models are convolutional neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *European Conference on Computer Vision*, 2014.

[13] J. Hu, W. Deng, and J. Guo, "Online regression of grandmother-cell responses with visual experience learning for face recognition," in *International Conference on Pattern Recognition*, 2014.

[14] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[15] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.

[16] C.-C. J. Kuo, "Understanding convolutional neural networks with a mathematical model," *Journal of Visual Communication and Image Representation*, vol. 41, pp. 406–413, 2016.

[17] C.-C. J. Kuo, "The CNN as guided multi-layer RECOS transform," to appear in *IEEE Signal Processing Magazine*, May 2017.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Neural Information Processing Systems*, 2012.

[19] Q. Le, M. Ranzato, R. Monga, M. Devin, K. Chen, G. Corrado, J. Dean, and A. Ng, "Building high-level features using large scale unsupervised learning," in *International Conference in Machine Learning*, 2012.

[20] Y. Li, L. Liu, C. Shen, and A. van den Hengel, "Mid-level deep pattern mining," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[21] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[22] R. Q. Quiroga, L. Reddy, G. Kreiman, C. Koch, and I. Fried, "Invariant visual representation by single neurons in the human brain," *Nature*, 2005.

[23] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Neural Information Processing Systems*, 2015.

[24] N. L. Roux, P.-A. Manzagol, and Y. Bengio, "Topmoumoute online natural gradient algorithm," in *Neural Information Processing Systems*, 2008.

[25] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei., "Imagenet large scale visual recognition challenge," in *International Journal of Computer Vision*, 2015.

[26] K. Simonyan, A. Vedaldi, and A. Zisserman, "Learning local feature descriptors using convex optimisation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014.

[27] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2014.

[28] R. K. Srivastava, J. Masci, F. Gomez, and J. Schmidhuber, "Understanding locally competitive networks," *ArXiv preprint ArXiv: 1410.1165*, 2015.

[29] C. Szegedy, W. Liua, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going dedeep with convolutions," in *IEEE Converence on Computer Vision and Pattern Recognition*, 2015.

[30] T. Tieleman and G. E. Hinton, "Neural networks for machine learning," tech. rep., University of Toronto, 2012.

[31] A. Torralba and A. Efros, "Unbiased look at dataset bias," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[32] D. Wei, B. Zhou, A. Torralba, and W. Freeman, "Understanding intra-class knowledge inside cnn," *ArXiv preprint ArXiv: 1507.02379*, 2015.

[33] H. Wersing and E. Korner, "Learning optimized features for hierarchical models of invariant object recognition," *Neural Computation*, 2003.

[34] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, "Understanding neural networks through deep visualization," in *Deep Learning Workshop of International Conference on Machine Learning*, 2015.

[35] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European Conference on Computer Vision*, 2014.

[36] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Object detectors emerge in deep scene cnns," in *International Conference on Learning Representations*, 2015.