Multifaceted Feature Visualization: Uncovering the Different Types of Features Learned By Each Neuron in Deep Neural Networks

Anh Nguyen
University of Wyoming

ANGUYEN8@UWYO.EDU

Jason Yosinski
Cornell University and Geometric Intelligence

JASON@GEOMETRICINTELLIGENCE.COM

Jeff Clune
University of Wyoming

JEFFCLUNE@UWYO.EDU

Abstract

We can better understand deep neural networks by identifying which features each of their neurons have learned to detect. To do so, researchers have created Deep Visualization techniques including *activation maximization*, which synthetically generates inputs (e.g. images) that maximally activate each neuron. A limitation of current techniques is that they assume each neuron detects only one type of feature, but we know that neurons can be *multifaceted*, in that they fire in response to many different types of features: for example, a grocery store class neuron must activate either for rows of produce or for a storefront. Previous activation maximization techniques constructed images without regard for the multiple different facets of a neuron, creating inappropriate mixes of colors, parts of objects, scales, orientations, etc. Here we introduce an algorithm that explicitly uncovers the multiple facets of each neuron by producing a synthetic visualization of each of the types of images that activate a neuron. We also introduce regularization methods that produce state-of-the-art results in terms of the interpretability of images obtained by activation maximization. By separately synthesizing each type of image a neuron fires in response to, the visualizations have more appropriate colors and coherent global structure. Multifaceted feature visualization thus provides a clearer and more comprehensive description of the role of each neuron.

Reconstructions of multiple feature types (facets) recognized by the same "grocery store" neuron
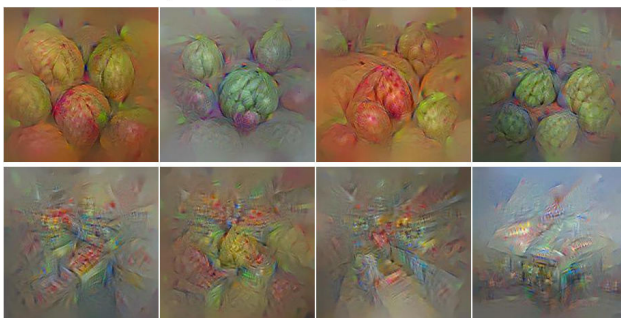
Corresponding example training set images recognized by the same neuron as in the "grocery store" class
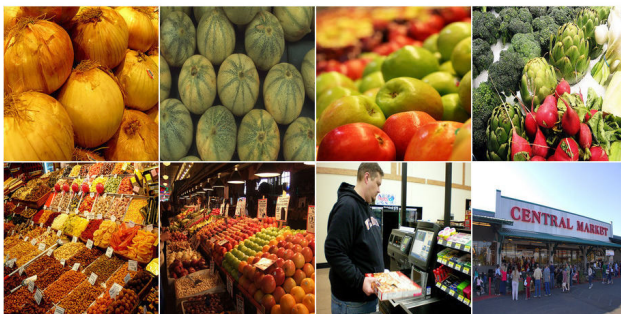

Figure 1. **Top:** Visualizations of 8 types of images (feature facets) that activate the same "grocery store" class neuron. **Bottom:** Example training set images that activate the same neuron, and resemble the corresponding synthetic image in the top panel.

1. Introduction

Recently, deep neural networks (DNNs) have demonstrated state-of-the-art—and sometimes human-competitive—results on many pattern recognition tasks, especially vision classification problems (Krizhevsky et al., 2012; Szegedy et al., 2014; He et al., 2015). That success has motivated efforts to better understand the inner workings

of such networks, which enables us to further improve their architectures, learning algorithms, and hyperparameters. An active area of research in this vein, called *Deep Visualization*, involves taking a trained DNN and creating synthetic images that produce specific neural activations of interest within it (Erhan et al., 2009; Simonyan et al., 2013; Yosinski et al., 2015; Dosovitskiy et al., 2015; Mahendran et al., 2015; Nguyen et al., 2015; Wei et al., 2015).

There are two general camps within Deep Visualization: *activation maximization* (Erhan et al., 2009) and *code inversion* (Mahendran et al., 2015). Activation maximization is the task of finding an image that maximally activates a certain neuron (aka "unit", "feature", or "feature detector"), which can reveal what each neuron in a DNN has learned to fire in response to (i.e. which features it detects). This technique can be performed for the output neurons, such as neurons that classify types of images (Simonyan et al., 2013), and can also be performed for each of the hidden neurons in a DNN (Erhan et al., 2009; Yosinski et al., 2015). Code inversion is the problem of synthesizing an image that, for a specific DNN layer, produces a similar activation vector at that layer as a target activation vector produced by a specific real image (Mahendran et al., 2015; Dosovitskiy et al., 2015). It reveals what information about one specific image is encoded by the DNN code at a particular layer.

Both activation maximization and code inversion start from a random image and calculate via backpropagation how the color of each pixel should be changed to either increase the activation of a neuron (for activation maximization) or produce a layer code closer to the target (for code inversion). However, previous studies have shown that doing so using only the gradient information produces unrealistic-looking images that are not recognizable (Simonyan et al., 2013; Nguyen et al., 2015), because the set of all possible images is so vast that it is possible to produce images that satisfy the objective, but are still unrecognizable. Instead, we must both satisfy the objective and try to limit the set of images to those that resemble natural images. Biasing optimization to produce more natural images can be accomplished by incorporating *natural image priors* into optimization, which has been shown to substantially improve the recognizability of the images generated. Many regularization techniques have been introduced that improve image quality such as: Gaussian blur (Yosinski et al., 2015), α-norm (Simonyan et al., 2013), total variation (Mahendran et al., 2015), jitter (Mordvintsev et al., 2015), and data-driven patch priors (Wei et al., 2015).

While these techniques have improved Deep Visualization methods over the last two years, the resultant images provide room for improvement: (1) the color distribution is unnatural (Fig. S3b-e); (2) recognizable fragments of images are repeated, but these fragments do not fit together into a

Figure 2. Visualizing the different facets of a neuron that detects bell peppers. Diverse facets include a single, red bell pepper on a white background (1), multiple red peppers (5), yellow peppers (8), and green peppers on the plant (4), a cutting board (6), or against a dark background (10). Center: training set Images from the bell pepper class are projected into two dimensions by t-SNE and clustered by k-means (see Sec. 2.3). Sides: synthetic images generated by multifaceted feature visualization for the "bell pepper" class neuron for each of the 10 numbered facets. Best viewed electronically, in color, with zoom. Larger version: Fig. S5. Fig. S6 shows similar results for the fishing reel class.

coherent whole: e.g. multiple ostrich heads without bodies, or eyes without faces (Fig. S3b) (Simonyan et al., 2013; Mahendran et al., 2015); (3) previous techniques have no systematic method to visualize different facets (types of stimuli) that a neuron respond to, but high-level neurons are known to be multifaceted. For example, a face-detecting neuron in a DNN was shown to respond to both human and lion faces (Yosinski et al., 2015). Neurons in human brains are similarly multifaceted: a "Halle Berry" neuron was found that responds to very different stimuli related to the actress, from pictures of her in costume to her name printed as text (Quiroga et al., 2005).

We name the class of algorithms that visualize different facets of a neuron *multifaceted feature visualization* (MFV). Erhan et al., 2009 found that optimizing an image to maximally activate a neuron from multiple random starting images usually yielded the same final visualization. In contrast, Wei et al., 2015 found that if the backpropagation neural pathway is masked out in a certain way, the resultant images can sometimes reveal different feature facets. This is the first MFV method to our knowledge; however, it was shown to visualize only two facets per output neuron, and is not able to systematically visualize all facets per neuron.

In this paper, we propose two Deep Visualization techniques. Most importantly, we introduce a novel MFV algorithm that: **(1)** Sheds much more light on the inner workings of DNNs than other state-of-the-art Deep Visualization methods by revealing the different facets of each neuron.

It also reveals that neurons at all levels are multifaceted, and shows that higher-level neurons are more multifaceted than lower-level ones (Fig. 6). **(2)** Improves the quality of synthesized images, producing state-of-the-art activation maximization results: the colors are more natural and the images are more globally consistent (Fig. 5. See also Figs. 1, 4, 2) because each facet is separately synthesized. For example, MFV makes one image of a green bell pepper and another of a red bell pepper instead of trying to simultaneously make both (Fig. 2). The increased global structure and contextual details in the optimized images also support recent observations (Yosinski et al., 2015) that DNNs encode not only knowledge of a sparse set of discriminative features for performing classification, but also more holistic information about typical input examples in a manner more reminiscent of generative models. **(3)** Is simple to implement. The only main difference is how activation maximization algorithms are initialized. To obtain an initialization per facet, we project the training set images that maximally activate a neuron into a low-dimensional space (here, a 2D space via t-SNE), cluster the images via k-means, and average the n (here, 15) closest images to each cluster centroid to produce the initial image (Section 3).

We also introduce a *center-biased regularization* technique that attempts to produce one central object. It combats a flaw with all activation maximization techniques including MFV: they tend to produce many repeated object fragments in an image, instead of objects with more coherent global structure. It does so by allowing, on average, more optimization iterations for center pixels than edge pixels.

2. Methods

2.1. Convolutional neural networks

For comparison with recent studies, we test our Deep Visualization methods on a variant of the well-known "AlexNet" convnet architecture (Krizhevsky et al., 2012), which is trained on the 1.3-million-image ILSVRC 2012 ImageNet dataset (Deng et al., 2009; Russakovsky et al., 2014). Specifically, our DNN follows the CaffeNet architecture (Jia et al., 2014) with the weights provided by Yosinski et al., 2015. While this network is slightly different than AlexNet, the changes are inconsequential: the network obtains a 20.1% top-5 error rate, which is similar to AlexNet's 18.2% (Krizhevsky et al., 2012).

The last three layers of the DNN are fully connected: we call them fc6, fc7 and fc8. fc8 is the last layer (before softmax transformation) and has 1000 outputs, one for each ImageNet class. fc6 and fc7 both have 4096 outputs.

2.2. Activation maximization

Informally, we run an optimization algorithm that optimizes the colors of the pixels in the image to maximally activate a particular neuron. That is accomplished by calculating the derivative of the target neuron activation with respect to each pixel, which describes how to change the pixel color to increase the activation of that neuron. We also need to incorporate natural image priors to bias the images to remain in the set of images that look as much as possible like natural (i.e. real-world) images.

Formally, we may pose the activation maximization problem for a unit with index j on a layer l of a network Φ as finding an image $\mathbf{x}^*$ where:

$$\mathbf{x}^* = \arg\max_{\mathbf{x}}(\Phi_{l,j}(\mathbf{x}) - R_\theta(\mathbf{x})) \tag{1}$$

Here, $R_\theta(\mathbf{x})$ is a parameterized regularization function that could include multiple regularizers (i.e. priors), each of which penalizes the search in a different way to collectively improve the image quality. We apply two regularizers: total variation (TV) and jitter, as described in Mahendran et al. (2015), but via a slightly different, but qualitatively similar, implementation. Supplementary Sec. S2 details the differences and analyzes the benefits of incorporating different regularizers used in previous activation maximization papers: Gaussian blur (Yosinski et al., 2015), α-norm (Simonyan et al., 2013), total variation (TV) (Mahendran et al., 2015), jitter (Mordvintsev et al., 2015), and a data-driven patch prior (Wei et al., 2015). Our code and parameters are available at http://www.EvolvingAI.org.

2.3. Multifaceted feature visualization

Although DNN feature detectors must recognize that very different types of images all represent the same concept (e.g. a bell pepper detector must recognize green, red, and orange peppers as in the same class), there is no systematic method for visualizing these different facets of feature detectors. In this section, we introduce a method for visualizing the multiple facets that each neuron in a DNN responds to. We demonstrate the technique on the ImageNet dataset, although the results generalize beyond computer vision to any domain (e.g. speech recognition, machine translation).

We start with the observation that in the training set, each ImageNet class has multiple intra-class clusters that reflect different facets of the class. For example, in the *bell pepper* class, one finds bell peppers of different colors, alone or in groups, cut open or whole, etc. (Fig. 2). We hypothesized that activation maximization on the bell pepper class is made difficult because which facet of the bell pepper to be reconstructed is not specified, potentially meaning that different areas of the image may optimize toward reconstructing different fragments of different facets. We further

---

Algorithm 1 Multifaceted Feature Visualization

    Input: a set of images U and a number of facets k

    1. for each image in U, **compute** high-level (here fc7) hidden code Φ_i

    2. **Reduce** the dimensionality of each code Φ_i from 4096 to 50 via PCA.

    3. **Run** t-SNE visualization on the entire set of codes Φ_i to produce a 2-D embedding (examples in Fig. 2).

    4. **Locate** k clusters in the embedding via k-means.

    for each cluster

        5. **Compute** a mean image $\mathbf{x_0}$ by averaging the 15 images nearest to the cluster centroid.

        6. **Run** activation maximization (see Section 2.2), but **initialize** it with $\mathbf{x_0}$ instead of a random image.

    Output: a set of facet visualizations $\{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_k}\}$.

---

hypothesized that if we initialize activation maximization optimization with the mean image of only one facet, that may increase the likelihood that optimization would reconstruct an image of that type. We also hypothesized that we can get reconstructions of different facets by starting optimization with mean images from different facets. Our experimental results support all of these hypotheses.

Specifically, to visualize different facets of an output neuron C (e.g. the fc8 neuron that declares an image to be a member of the "bell pepper" class), we take all ($\sim$1300) training set examples from that class and follow Algorithm 1 to produce $k = 10$ facet visualizations. Note that, here, we demonstrate the algorithm on the *training* set, and the results are shown in Figs. 1, 4, 2 & 5. However, we also apply the same technique on the *validation* set (see Sec. 3.2), and show its results in Fig. 6.

Intuitively, the idea is to (1) use a DNN's learned, abstract, knowledge of images to determine the different types of images that activate a neuron, and then (2) initialize a state-of-the-art activation maximization optimization algorithm with the mean image computed from each of these clusters to visualize each facet. For (1), we first embed all of the images in a class in a two-dimensional space via PCA (Person, 1901; Jolliffe, 2002) and t-SNE (Van der Maaten et al., 2008), and then perform k-means clustering (MacQueen et al., 1967) to find k types of images (Fig. 2). Note that here we only visualize 10 facets per neuron, but it is possible to visualize fewer or more facets by changing k. We compute a mean image by averaging $m = 15$ images (Algorithm 1, step 5) as it works the best compared to $m = \{1, 50, 100, 200\}$ (data not shown). Supplementary Sections S1 & S2.4 provide more intuition regarding why initializing from mean images helps.

2.4. Center-biased regularization

In activation maximization, to increase the activation of a given neuron (e.g. an "ostrich" neuron), each "drawing" step often includes two types of updates: (1) intensifying the colors of existing ostriches in the image, and (2) drawing new fragments of ostriches (e.g. multiple heads). Since both types of updates are non-separably encoded in the gradient, the results of previous activation maximization methods are often images with many repeated image fragments (Fig. S3b-h). To ameliorate this issue, we introduce a technique called *center-biased regularization*.

Preliminary experiments revealed that optimizing with a large smoothing effect (e.g. a high Gaussian blur radius or TV weight) produces a blurry, but single and centered object. Based on this observation, the intuition behind our technique is to first generate a blurry, centered object, and then refine this image by updating the center pixels more than the edge pixels to produce a final image that is sharp, and has a centrally-located object (Fig. 3).
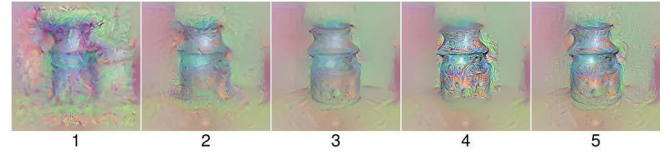

Figure 3. Progressive result of optimizing an image to activate the "milk can" neuron via the *Center-biased regularization* method.

Center-biased regularized images have far fewer duplicated fragments (Fig. S3i), and thus tend to more closely represent the style of training set images, which feature one centrally located object. However, this technique is not guaranteed to produce a single object only. Instead, it is a way of biasing optimization toward creating objects near the image center. Supplementary Section S2.3 provides more details and results for center-biased regularization, including how combining it with multifaceted visualization (initializing from mean images) further improves visualization quality.

3. Results

3.1. Neurons are multifaceted feature detectors

Multifaceted feature visualization produces qualitatively different images that activate the same neuron. For example, it synthesizes differently colored peppers for the "bell pepper" class neuron (Fig. 2) and differently colored cars for the "convertible" car class neuron (Fig. 4b). It also produces objects seen from different perspectives, such as cars seen from the back or front (Fig. 4b), or in different numbers, such as one, two, or multiple peppers (Fig. 2).

Most interestingly, multifaceted feature visualization can

uncover that a deep neural network has learned to recognize extremely different facets of a class, such as the following. From the movie theater class: the inside of the theater with rows of seats and a stage, and the external facade viewed at night or day (Fig. 4a); from the pool table class: zoomed in pictures of pool balls on green felt, a pool table against a white background, and pool tables in dark rooms (Fig. 4c); from the grocery store class: rows of differently colored produce, a scene with a cashier, and the facade of the store viewed from the parking lot (Fig. 1). See supplementary info (Fig. S4) for many more examples.

Somewhat surprisingly, some facets of a class are, after regularized optimization, actually classified as members of other classes. For example, many real images from the training set for the grocery store class feature rows of apples or artichokes, which should arguably instead be in the apple or artichoke classes (Fig. 1, bottom row). DNNs may learn to assign a high probability to both grocery store and apple for such images, and thus correctly classify the image under the "top-5" classification accuracy metric (Krizhevsky et al., 2012). Multifaceted feature reconstruction correctly recognizes that different facets of the grocery store class involve each of these types of produce, and renders separate synthetic images resembling rows of apples, artichokes, or oranges. As with the real training set images, the DNN labels such synthetic images as both "grocery store" and the type of produce. If optimization were carried out without regularization, then following the gradient toward the grocery store class would surely assign the highest probability to that class, considering how easy it is to steer the network to produce arbitrary outputs when that is the only cost (Szegedy et al., 2013; Nguyen et al., 2015; Goodfellow et al., 2014). However, here, when the optimization process is regularized, and initialized from the centroid of a cluster of training set images, the DNN is sometimes more confident that an image synthesized to light up its "grocery store" class is a member of another class (e.g. the "apple" class) than a member of the "grocery store" class! For example, visualizations of different facets of the "grocery store" neuron are most confidently labeled by the DNN as in the "custard apple" and "fig" classes (Fig. 1, top row, leftmost two images), and only secondly labeled as in the grocery store class. These results reinforce that MFV automatically discovers and reveals the multifaceted nature of images in a class.

3.2. Visualizing the multifaceted nature of hidden neurons

In addition to visualizing the class output neurons (layer fc8 in this DNN), we can also perform multifaceted feature visualization for neurons on hidden layers. To do so, we collect the top $n = 2\%$ (i.e. 1000) images (or image patches, depending on the receptive field

(a) *Movie theater*: outside (day & night) and inside views.

(b) *Convertible*: with different colors and both front & rear views.

(c) *Pool table*: Up close & from afar, with different backgrounds.

Figure 4. Multifaceted visualization of fc8 units uncovers interesting facets. We show 4 different facets for each neuron. In each pair of images, the bottom is the facet visualization that represents a cluster of images from the training set, and the top is the closest image to the visualization from the same cluster.

size of the neuron) from the $50,000$-image *validation set* that most highly activate the neuron, which become the set of images U in Algorithm 1. To determine the best n, we also experimented with a sweep of lower $n = \{0.05\%, 0.1\%, 0.15\%, 0.2\%, 0.25\%, 0.3\%\}$, but did not observe qualitative improvements (data not shown). To visualize a unit in a fully-connected layer l (e.g. fc6), we modify step 1 of Algorithm 1 to compute the code at l for each image in U (e.g. a vector of 4096 numbers, one for each fc6 neuron). For a convolutional layer unit (e.g. conv3), the layer code is the column of features (across all channels) at a given spatial location (e.g. the code for a conv3 unit at location $(5, 7)$ is a vector of 384 numbers,

one for each channel).

Fig. 6 shows multiple facets reconstructed for example hidden units of every layer. Because the overall quality of these visualizations is improved (discussed more below), it becomes easier to learn what each of these hidden neurons detects. Low-level layers (e.g. conv1 and conv2) do not exhibit noticeably different facets. However, starting at conv3, the qualitative difference between facets increases: first in slight differences of rotation, pose, and color, then increasingly to the number of objects present, and ultimately to very different types of objects (e.g. a closeup of an ostrich face vs. a pair of ostriches viewed from afar). This result mirrors the known phenomenon whereby features in DNNs become more abstract at higher layers, meaning that they are increasingly invariant to larger and larger changes in the input image (Mahendran et al., 2015; Bengio et al., 2015).

Another previously reported result that can be seen even more clearly in these improved visualizations is that neurons in convolutional layers often represent only a single, centered object, whereas neurons in fully connected layers are more likely to contain multiple objects (Yosinski et al., 2015). Interestingly, and not previously reported to our knowledge, neurons in hidden, fully connected layers often seem to be an amalgam of very different, high level concepts. For example, one neuron in fc6 looks like a combination of a scuba diver and a turtle (Fig. 6, leftmost fc6 neuron). In fact, within the top 15 images that activate that neuron, there are indeed pictures of turtles and scuba divers, but also whales and sharks. Perhaps this neuron is best described as a "something underwater" neuron. This result could occur because our facets are not entirely pure (perhaps with a higher k or different optimization objectives, we would end up with separate t-SNE clusters for underwater turtles, scuba divers, whales, and sharks). An alternate, but not mutually exclusive, explanation is that these feature detectors are truly amalgams, or at least represent abstract concepts (such as "something underwater"), because they are used to classify many different types of images. It is not until the final layer that units should classify specific types of objects, and indeed our visualizations show that these last-layer (fc8) class neurons are more pure (showing different facets of a single concept).

To further investigate these two hypotheses, we visualized the input patterns responsible for fc6 and fc7 neuron activations via two non-stochastic methods: Deconv (Zeiler et al., 2014) and Layer-wise Relevance Propagation (LRP) (Bach et al., 2015a). However, these methods did not reveal whether these units are truly amalgams (Sec. S3). Future research into these questions is necessary.

Fully connected hidden layer neuron reconstructions also revealed cases when, given totally different initialization

images (e.g. of a white photocopier and a truck), optimization still converges to the same concept (e.g. "a yellow dog") (Sec. S3 & Fig. S9). One might think that adding a L_2 penalty for deviating from the mean image in high-level (e.g. fc7) code space might prevent such convergence and visualize more facets. Our preliminary experiments with this idea did not produce improvements, but we will continue to investigate it in future work.

3.3. Multifaceted feature visualization improves the state of the art of activation maximization

Beside uncovering multiple facets of a neuron, our technique also improves over the previous state-of-the-art *activation maximization* methods. Figs. 5 & S3 showcase multifaceted feature visualization on many classes and compare it to previous methods. Note that these two figures only show one facet per class, specifically the largest among 10 clusters (i.e. Algorithm 1, with $k = 10$). While that cluster may be the largest because it contains the most canonical images, it may also be large because it is amorphous and k-means clustering could not subdivide it. Thus, our results might be even better when visualizing the most visually homogenous facet, which we will investigate in future work.

The resulting images have a substantially more natural color palette than those produced by previous algorithms (Figs. 5 & S3). That is especially evident in the background colors, which rarely change for other methods from class to class. In contrast, with multifaceted feature visualization, it is obvious when an image is set underwater or beneath a clear, blue sky. An additional improvement is that the images are more globally consistent, whereas previous methods had the problem of frequently repeated fragments of images that did not form a coherent whole.

Initializing from mean images may work because it biases optimization towards the general color layout of a facet, providing some global structure (Fig. S8 shows example mean images). An alternate hypothesis is that the averaging operation leaves a translucent version of all n original images, and optimization recovers one or several of them. Indeed, when initializing with interpolated averages between two images, optimization snaps to one or the other, instead of merging both (Fig. S1). However, these reconstructions snap to the overall facet *type* of the seed image— meaning the color scheme, global structure, and theme (e.g. ostriches on a grassy plain)—but are not faithful to the details of the image (e.g. the number of ostriches). The effect is more pronounced when averaging over 15 images: optimization often ignores the dominant object outlines that remain, and instead fills in different details (e.g. at a different scale), but in a way that still fits the context of the overall color layout (e.g. Fig. S8; note the cheeseburger and milk

can are smaller than the mean image suggests). Such observations are not conclusive, however, and our paper motivates future research into the precise dynamics of why initializing with a mean facet image improves the quality of activation maximization.

4. Discussion and Conclusion

One way to study a neuron's different feature facets is to simply perform t-SNE on the real images that maximally activate it (Fig. 2). However, doing so does not reveal what a neuron knows about a concept or class. Based on 'fooling' images, scientists previously concluded that DNNs trained with supervised learning ignore an object's global structure, and instead only learn a few, discriminative features per class (e.g. color or texture) (Nguyen et al., 2015). Our work here strengthens later findings (Yosinski et al., 2015; Mahendran et al., 2015) showing that, in contrast, DNNs trained with supervised learning act more like generative models by learning the global structure, details, and context of objects. They also learn their multiple facets.

Activation maximization can also reveal what a DNN ignores when classifying. A reason that our method does not always reconstruct a proper facet, e.g. of *stuffed* peppers (Fig. 2, cluster 7), could be that the bell pepper neuron ignores, or lightly weights, stuffing. The number of unique images MFV produces for a unit depends on the k in k-means (here, manually set). Automatically identifying the true number of facets is an important, open scientific question raised, but not answered, by this paper.

Overall, we have introduced a simple Multifaceted Feature Visualization algorithm, which (1) improves the state of the art of activation maximization by producing higher quality images with more global structure, details, context, more natural colors, and (2) shows the multiple feature facets each neuron detects, which provides a more comprehensive understanding of each neuron's function. We also introduced a novel *center-biased regularization* technique, which reduces optimization's tendency to produce repeated object fragments and instead tends to produce one central object. Such improved Deep Visualization techniques will increase our understanding of deep neural networks, which will in turn improve our ability to create even more powerful deep learning algorithms.

Figure 5. Visualizations of the facet with the most images for 50 example fc8 class neurons that showcase realistic color distributions and globally consistent objects. While subjective, we believe the improved color, detail, global consistency, and overall recognizability of these images represents the state of the art in visualization using activation maximization (Fig. S3). Moreover, the improved quality of the images reveals that even the highest-level neurons in deep neural networks encode much more information about classes than was previously thought, such as the global structure, details, and context of objects. Best viewed in color with zoom. Fig. S2 shows many more examples.

Figure 6. Multifaceted visualization of example neuron feature detectors from all eight layers of a deep convolutional neural network. The images reflect the true sizes of the receptive fields at different layers. For each neuron, we show visualizations of 4 different facets. These images are hand picked to showcase the diverse neurons and their multifaceted nature. Neurons in layers 1 and 2 do not have noticeably different facets. However, starting from layer 3, the visualizations reveal that neurons at higher layers are increasingly complex and have diverse facets. This increased facet diversity is because higher level features are more invariant to large changes in the image, such as color, number, pose, and context. Interestingly, units in layer 6 and 7 seem to blend different objects together. For example, the visualization of the leftmost layer 6 neuron seemingly combines a turtle and a scuba diver. It turns out this neuron responds to images of "something underwater", including turtles and scuba divers, but also whales and sharks (Fig. S9c). This is likely because such neurons are involved in a distributed code of abstract concepts that exist in many different classes of images. At the final, 8th, layer, where neurons are trained to respond separately for each class, the facets are still diverse, but are within a class. Fig. S4 shows many more examples for fc8. Best viewed electronically, in color, with zoom.

References

Bach, Sebastian, Binder, Alexander, Montavon, Grégoire, Klauschen, Frederick, Müller, Klaus-Robert, and Samek, Wojciech. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7), 2015a. 6, 20, 22

Bach, Sebastian, Binder, Alexander, Montavon, Grégoire, Müller, Klaus-Robert, and Samek, Wojciech. Lrp toolbox for artificial neural networks 1.0–manual. 2015b. 20

Bengio, Yoshua, Goodfellow, Ian J., and Courville, Aaron. Deep learning. Book in preparation for MIT Press, 2015. URL http://www.iro.umontreal.ca/~bengioy/dlbook. 6

Deng, Jia, Dong, Wei, Socher, Richard, Li, Li-Jia, Li, Kai, and Fei-Fei, Li. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 248–255. IEEE, 2009. 3

Dosovitskiy, Alexey and Brox, Thomas. Inverting visual representations with convolutional networks. *arXiv preprint arXiv:1506.02753*, 2015. 2

Erhan, Dumitru, Bengio, Yoshua, Courville, Aaron, and Vincent, Pascal. Visualizing higher-layer features of a deep network. *Dept. IRO, Université de Montréal, Tech. Rep*, 4323, 2009. 2

Getreuer, Pascal. Rudin-osher-fatemi total variation denoising using split bregman. *Image Processing On Line*, 10, 2012. 12

Goldstein, Tom and Osher, Stanley. The split bregman method for l1-regularized problems. *SIAM Journal on Imaging Sciences*, 2(2):323–343, 2009. 12

Goodfellow, Ian J, Shlens, Jonathon, and Szegedy, Christian. Explaining and Harnessing Adversarial Examples. *ArXiv e-prints*, December 2014. 5

He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, and Sun, Jian. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015. 1

Jia, Yangqing, Shelhamer, Evan, Donahue, Jeff, Karayev, Sergey, Long, Jonathan, Girshick, Ross, Guadarrama, Sergio, and Darrell, Trevor. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. 3

Jolliffe, Ian. *Principal component analysis*. Wiley Online Library, 2002. 4

Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012. 1, 3, 5

MacQueen, James et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pp. 281–297. Oakland, CA, USA., 1967. 4

Mahendran, Aravindh and Vedaldi, Andrea. Visualizing deep convolutional neural networks using natural pre-images. *arXiv preprint arXiv:1512.02017*, 2015. 2, 3, 6, 7, 12, 13, 14, 16, 19

Mordvintsev, Alexander, Olah, Christopher, and Tyka, Mike. Inceptionism: Going deeper into neural networks. *Google Research Blog. Retrieved June*, 20, 2015. 2, 3, 13

Nguyen, Anh, Yosinski, Jason, and Clune, Jeff. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proc. of the Conference on Computer Vision and Pattern Recognition*, 2015. 2, 5, 7, 11

Person, K. On lines and planes of closest fit to system of points in space. philiosophical magazine, 2, 559-572, 1901. 4

Quiroga, R Quian, Reddy, Leila, Kreiman, Gabriel, Koch, Christof, and Fried, Itzhak. Invariant visual representation by single neurons in the human brain. *Nature*, 435 (7045):1102–1107, 2005. 2

Russakovsky, Olga, Deng, Jia, Su, Hao, Krause, Jonathan, Satheesh, Sanjeev, Ma, Sean, Huang, Zhiheng, Karpathy, Andrej, Khosla, Aditya, Bernstein, Michael, et al. Imagenet large scale visual recognition challenge. *arXiv preprint arXiv:1409.0575*, 2014. 3

Samek, Wojciech, Binder, Alexander, Montavon, Grégoire, Bach, Sebastian, and Müller, Klaus-Robert. Evaluating the visualization of what a deep neural network has learned. *arXiv preprint arXiv:1509.06321*, 2015. 20

Simonyan, Karen, Vedaldi, Andrea, and Zisserman, Andrew. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013. 2, 3, 16

Strong, David and Chan, Tony. Edge-preserving and scale-dependent properties of total variation regularization. *Inverse problems*, 19(6):S165, 2003. 11

Szegedy, Christian, Zaremba, Wojciech, Sutskever, Ilya, Bruna, Joan, Erhan, Dumitru, Goodfellow, Ian J., and

Fergus, Rob. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013. 5

Szegedy, Christian, Liu, Wei, Jia, Yangqing, Sermanet, Pierre, Reed, Scott, Anguelov, Dragomir, Erhan, Dumitru, Vanhoucke, Vincent, and Rabinovich, Andrew. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014. 1

Van der Maaten, Laurens and Hinton, Geoffrey. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2579-2605):85, 2008. 4

Wei, Donglai, Zhou, Bolei, Torrabla, Antonio, and Freeman, William. Understanding intra-class knowledge inside cnn. *arXiv preprint arXiv:1507.02379*, 2015. 2, 3, 14, 16

Yosinski, Jason, Clune, Jeff, Nguyen, Anh, Fuchs, Thomas, and Lipson, Hod. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015. 2, 3, 6, 7, 11, 13, 14, 16, 20

Zeiler, Matthew D and Fergus, Rob. Visualizing and understanding convolutional networks. In *Computer Vision– ECCV 2014*, pp. 818–833. Springer, 2014. 6, 20, 22

Supplementary Information for:
Multifaceted Feature Visualization: Uncovering the Different Types of Features Learned By Each Neuron in Deep Neural Networks

Anh Nguyen   ANGUYEN8@UWYO.EDU
Jason Yosinski   JASON@GEOMETRICINTELLIGENCE.COM
Jeff Clune   JEFFCLUNE@UWYO.EDU

S1. Activation maximization initialized with interpolated images

We investigated whether optimization starting from a mean image that is the average of two different types of images (i.e. two different facets) would produce either (a) a visualization that resembles a mix of the two facets, or (b) a reconstruction of only one of the facets, which could occur if one facet win outs and optimization gravitates toward it, erasing the other facet.

We first take all ~1,300 of the images in a training set class (e.g. the ostrich class) and run Algorithm 1 (step 1-4) with $k = 10$ to produce 10 clusters of images. We select two random images, each from a different cluster, and generate a series of 8 intermediate linearly interpolated images between those two images (e.g. in the top row of Fig. S1a, the leftmost and rightmost are two training set images). We then run activation maximization with the same parameters as in the multifaceted visualization experiments in the main text (Sec. 2.3), with the initialization being each of the interpolated images. Each panel in Fig. S1 shows a series of interpolated images (top row) and the corresponding activation maximization results (bottom row).

We found that when starting from a single real image, sometimes optimization produces a visualization that fits in the same facet as the initial image, but with different details. For example, in Fig. S1a, the leftmost and rightmost synthetic images are started with the (non-modified) training set image above them and reproduce a similar *type* of image (large ostrich heads against a blue sky or ostriches on a grassy plain from afar), but the details differ. Because of regularization (TV and jitter) and optimization, activation maximization completely redraws the main subject, and actually produces more than one ostrich in both cases (zoom in to see more easily).

In other cases, optimization does not take the guide from the initial image, but instead converges to a different facet altogether. For example, when seeded with either a stuffed or cut-open pepper, optimization instead seemingly pro-duces a whole pepper (Fig. S1c). This could happen because a "stuffed pepper" facet and a regular "whole pepper" facet share a lot of common details (reflective skin, colors), thus, the bias in the seed image might not be strong enough to pull the optimization toward the stuffed pepper facet. Another hypothesis is that the DNN has never learned to associate the stuffing detail with the bell pepper concept. A final hypothesis is that whole peppers are much more common, making their basin of attraction much larger and the gradients toward them steeper.

Overall, we observe that optimization often reconstructs one facet or another, but not a hybrid of both (Figs. S1a & S1b). Interestingly, and for unknown reasons, in the fishing reel example, optimization did not produce the 'fisherman against water and sky from afar' facet until the 3rd interpolated image (Fig. S1b, 3rd image from the left). Instead, in the leftmost two images of Fig. S1b, optimization produced a totally different, and common, facet of a close-up of a fishing reel (Fig. S6, most facets).

S2. Comparison between different priors

Many regularizers have been proposed to improve the quality of activation maximization images. Here, we compare the benefits of incorporating some of the leading regularizers from previous activation maximization papers.

S2.1. Total variation

Optimizing images via gradient descent to maximize the activation of a neuron only, without any regularization, produces unrecognizable images with overly high frequency information (Yosinski et al., 2015; Nguyen et al., 2015). Yosinski et al. (2015) proposed incorporating Gaussian blur with a small radius to smooth out the image. While this technique improves recognizability, it causes an overly blurry image because sharp edges are blurred away (Fig. S3c). Total variation (TV) regularization is a different smoothing technique that combats this issue by minimizing the total variation across adjacent pixels in an image while preserving the sharpness of edges (Strong et al., 2003). In
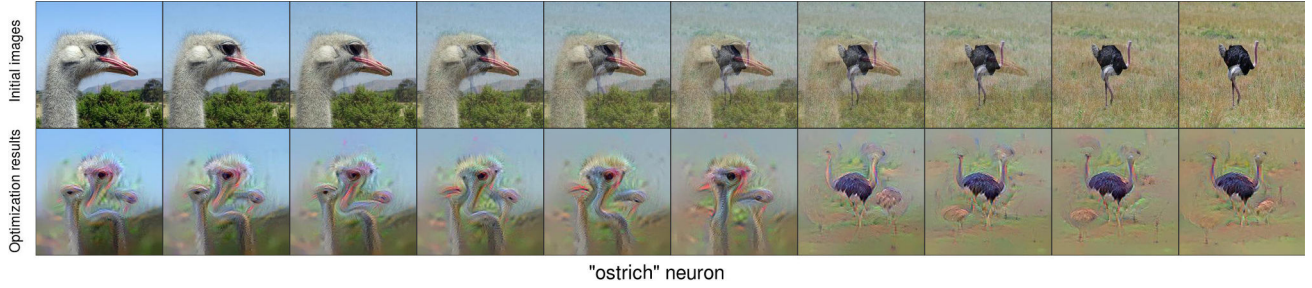
(a) Optimization is pulled toward one facet or the other instead of visualizing a combination of both. Interestingly, even when optimization is started from a single image (left and right extremes), optimization and regularization combine to produce an image in the same style (ostrich heads against a blue sky or ostriches on a grassy plain from afar), but with different details (e.g. the number of ostriches).
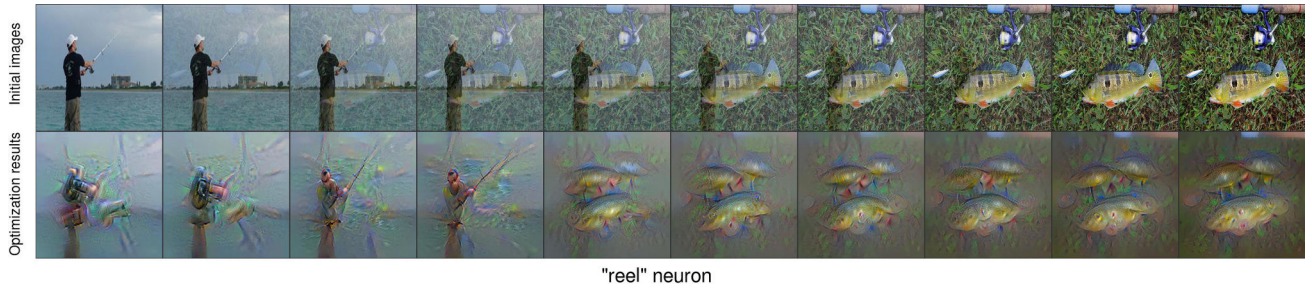
(b) Optimization either draws a fisherman against water and sky or a fish on grass. Interestingly, the fish images, which are generated to maximize the reel class, add additional fish to the one in the seed image, instead of adding fishing reels.
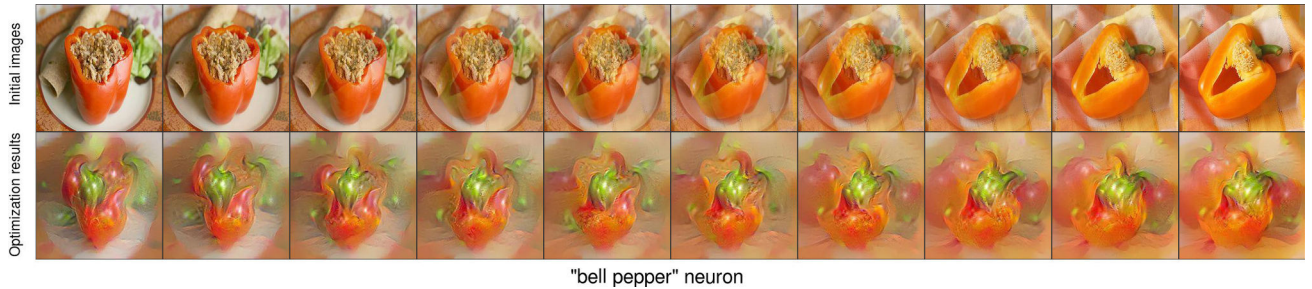
(c) Optimization seems to converge to *neither* of the two facets, and instead produces a whole pepper (with inconsistent coloring). See text in Sec. S1 for hypothesis as to why this may occur.

Figure S1. Optimization seeded with interpolated real images (top rows) often reconstructs either one facet or the other, but not a hybrid of both (bottom rows).

combination with other regularizers such as α-norm, this prior is effective in both code inversion and activation maximization tasks (Mahendran et al., 2015).

While Mahendran et al. (2015) incorporated the TV norm as a penalty in the objective function (Eq. 1), here we solve the TV minimization problem separately from the activation maximization problem. Specifically, each of our optimization iterations has two steps: (1) update the image $\mathbf{x}$ in the direction that maximizes the activation of a given neuron; (2) find an image that is closest to $\mathbf{x}$ that has the smallest TV via an iterative split Bregman optimization (Goldstein et al., 2009). Split Bregman is an algorithm that is designed to flexibly solve non-differential convex minimiza-

tion problems by splitting the objective terms, and is especially efficient for TV regularization (Getreuer, 2012). For each iteration of maximizing the neural activation, we perform 100 iterations of minimizing TV. Empirically, we found this strategy more flexible, which allowed us to discover slightly better results than the method in Mahendran et al., 2015.

In extensive preliminary activation maximization experiments (data not shown), we found that fairly good visualizations emerge by applying TV regularization *only*. Our results with TV alone (Fig. S3g) were not qualitatively improved by adding jitter (Fig. S3e), and were better than methods that predate Mahendran et al., 2015 (Fig. S3b-d).

S2.2. Jitter

An optimization regularization technique called "jitter" was first introduced by Mordvintsev et al., 2015 and later used in Mahendran et al., 2015. The method involves: (1) creating a canvas (e.g. of size 272×272) that is larger than the DNN input size (227×227) and (2) iteratively optimizing random 227×227 regions on the canvas. Optimization with jitter often results in high-resolution, crisp images; however, it does not ameliorate the problems of unnatural coloration and the repetition of image fragments that do not form a coherent, sensible whole.

Because we believe it represents the best previous activation maximization technique, we reproduce the algorithm of Mahendran et al. (2015), which applies both jitter and TV regularizers (Figs. S8a & S3e). Mahendran et al. (2015) report that TV is the most important prior in their framework. We found that Gaussian blur works just as well if combined with jitter while gradually reducing the blurring effect (i.e. radius) during optimization (Fig. S3h). This combination works because: (1) as shown in Yosinski et al. (2015), the smoothing effect by Gaussian blur enables optimization to find better local optima that have more global structure; (2) reducing the blur radius over time minimizes the blurring artifact of this prior in the final result; and (3) jitter further enhances the sharpness of images.

S2.3. Center-biased regularization

Previous activation maximization methods produce images with many unnatural repeated image fragments (Fig. S3b-h). Though to a lesser degree, multifaceted feature visualization also produces such repetitions (Fig. S3k). Such repetitions are not found in the vast majority of training set images. For example, a canonical training set image in the "beacon" class often shows a single lighthouse (Fig. S3a); however, Deep Visualization techniques show many more beacons or patches of beacons (Fig. S3b-h, k). To ameliorate this issue, in this section we introduce a technique called *center-biased regularization*.

Our method builds upon the idea of combining TV (Sec. S2.1) and jitter (Sec. S2.2) regularizers, following (Mahendran et al., 2015), and adds an additional bias towards a restricted "drawing" region in the center of the image. In a preliminary experiment (data not shown), we found that optimization with a large smoothing effect (e.g. a high Gaussian blur radius or TV weight) often results in a blurry, but single and centered object in the visualization. Based on this observation, the intuition behind our technique is to first generate a blurry, centered-object image (Fig. S7, leftmost image), and then optimize the center pixels more than the edge pixels to produce a final image that is sharp and has a centrally-located (Fig. S7, 4 right images). Multiple examples for different classes are shown

in Fig. S8b.

Figure S7. Progressive result of optimizing an image to activate the "milk can" neuron via *center-biased regularization*. Each image is the result of one out of five optimization phases. This figure is also shown in the main text (Fig. 3).

OPTIMIZATION SCHEDULE AND PARAMETERS

Specifically, our optimization schedule is carefully tuned with five phases (Fig. S7), where each has a different set of parameters. As described in Sec. S2.1, each of our optimization iterations has two steps: (1) update the image $\mathbf{x}$ in the direction that maximizes the activation of a given neuron; (2) find a smoothed image $\mathbf{x_s}$ that is closest to $\mathbf{x}$ that has the smallest TV. This $\mathbf{x_s}$ will become the initial image $\mathbf{x}$ in the next optimization iteration. We begin by describing the first three phases that are the most important, each of which runs for 150 iterations.

Phase 1-3: First, to generate a blurry image, we start with a low L_2 regularization parameter $\lambda = 0.001$ when finding a smoothed image $\mathbf{x_s}$ (step 2). A large λ forces $\mathbf{x_s}$ to be close to $\mathbf{x}$, and results in a sharp image; while a small λ allows $\mathbf{x_s}$ to be far from $\mathbf{x}$, and results in a smoothed image. Specifically, for the first 3 phases, $\lambda = \{0.001, 0.08, 2\}$. We also use a lower learning rate for each phase when updating image $\mathbf{x}$ in the activation maximization direction (step 1): $11, 6, 1$. The intuition is to force the optimization to lock in on the object (e.g. a milk can) that appears in phase 1 (Fig. S7, leftmost). In other words, we try to minimize the chance of new duplicated fragments of milk cans to appear toward the end of the optimization as seen in Fig. S8a.

To bias the main object to appear in the center, for each phase, we increase the canvas size (i.e. upsampling the image $\mathbf{x}$ by 20% at the beginning of each phase) to be: 227×227, 272×272, and 327×327. The regular jittering approach involves sampling and sending a random 227×227 patch (anywhere across the canvas) to the DNN for optimization. Here, we restrict such sampling so that the center of a patch is within a small canvas-centered square. By the end of phase 3, the visualization often has a single, centered object (Fig. S7, phase 3).

Phase 4-5: The purpose of phase 4 and 5 is to sharpen the centered object without generating new duplicated fragments. We attempt to do this by center-cropping the gradient image (i.e. the gradient backpropagated from the DNN has the form of a $3 \times 227 \times 227$ image) down to $3 \times 127 \times 127$. In addition, in phase 4, we restrict the op-

timization to appear in the center of the image only, and optimize for 30 iterations. As the result, our object has a sharpened region in the center (Fig. S7, phase 4). Finally, to balance out the sharpness between the center and edge pixels, in phase 5, we optimize for 10 iterations while allowing jittering to occur anywhere in the image. Thus, the final visualization is often a sharper version of the phase 3 result (Fig. S7, phase 5 vs phase 3).

The center-biased regularized images have far fewer duplicated fragments (Fig. S3i), thus, tend to more closely represent the style of training set images, which feature one centrally located object. However, this technique is not guaranteed to produce a single object only. Instead, it is a way of biasing optimization toward creating objects near the image center. Sec. S2.4 shows the experiment combining center-biased regularization and multifaceted visualization (initializing from mean images) to further improve the visualization quality.

S2.4. Initialization with mean images

An issue with previous activation maximization methods is that the images tend to have an unrealistic color distribution (Fig. S3b-c, e-h). A straightforward approach to ameliorate this problem is to incorporate an α-norm regularizer, which encourages the intensity of pixels to stay within a given range (Mahendran et al., 2015; Yosinski et al., 2015). While this method is effective in suppressing extreme color values, it does not improve the realism of the overall color distribution of images. Wei et al., 2015 proposed a more advanced data-driven patch prior regularization to enforce the visualizations to match the colors of a set of natural image patches. While this prior substantially improved the colors for code inversion, its results for activation maximization still have several issues (as seen in Fig. S3d): (1) having duplicated fragments (e.g. duplicated patches of lighthouses in a "beacon" image), and (2) lacking details, producing unnatural images.

Here, our multifaceted visualization improves the color distribution of images via a different approach: starting optimization from a mean image (Fig. S8d) computed from real training examples (see Sec. 3.3). A possible explanation for why this works is that initializing from the mean image puts the search in a low-dimensional manifold that is much closer to that of real images, and thus it is easier to find a realistic looking image around this area. The mean image thus provides a general outline for a type of image based on blurry colors and optimization can fill in the details. For example, the center-biased regularization technique was able to produce a single milk can, but did so without a relevant contextual setting (Fig. S8b). However, when this technique is initialized with a mean image (computed from 15 images from the "milk can" class) that has

a blurry brown surface (Fig. S8d, milk can), optimization turned that general layout into a complete, coherent picture: a milk can on a table (Fig. S8c). The colors of the visualizations are also substantially improved when optimization is initialized from mean images (Fig. S8b versus c).

We observed that multifaceted visualization images often exhibit centered objects (Fig. 1 & 4), and thus we found no substantial qualitative improvement when adding center-biased regularization to it (Fig. S8c). That could be because the mean image provides enough of a global layout for the style of the image, which in ImageNet is usually a single, centered object.

While simple, our technique results in images that have qualitatively more realistic colors than previous methods (Figs. S2 & S3k). This technique is also expected to work with any dataset (e.g. grayscale images, or images of a special topic). Note that our method does not require access to the training set. If the training set is unavailable, one can simply pass any natural images (or other modes of input such as audio if not reconstructing images) to get a set of images that highly activate a neuron. A similar idea was used in Wei et al. (2015), who built an external dataset of patches that have similar characteristics to the DNN training set.
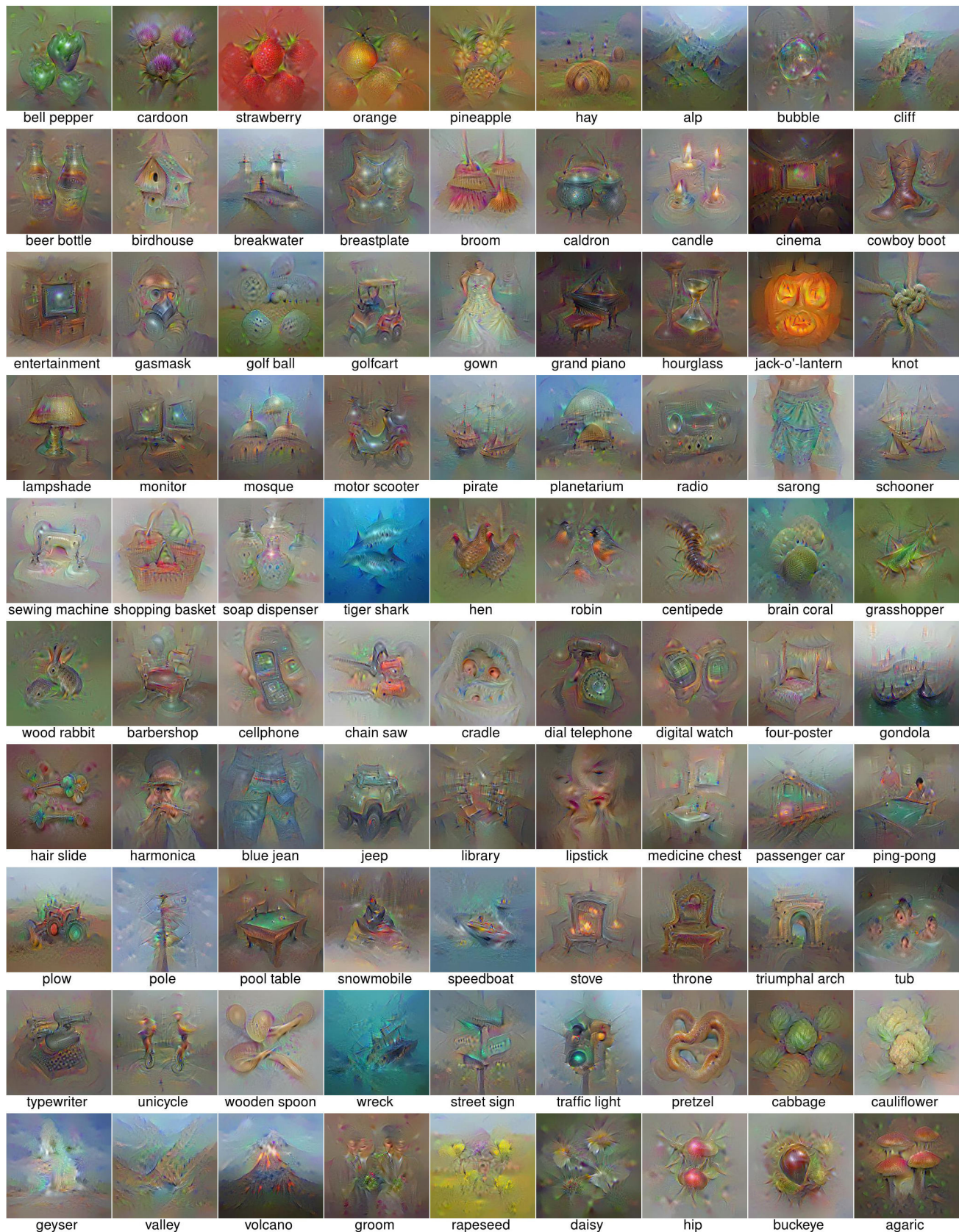
Figure S2. Visualizations for 90 example fc8 class neurons of the facet that has the largest number of images (Fig. S4 shows examples with four facets each). We selected classes that showcase the realistic color distribution of images and global consistency of objects. While evaluating images is subjective, we believe the improved color, global consistency, and overall recognizability of these images represents the state of the art in visualization using activation maximization (Fig. S3 offers a side-by-side comparison to previous techniques). Moreover, the improved quality of the images reveals that even at the highest-level layer, deep neural networks still encode much more information about classes than was previously thought. Best viewed in color. This is an expanded version of Fig. 5.
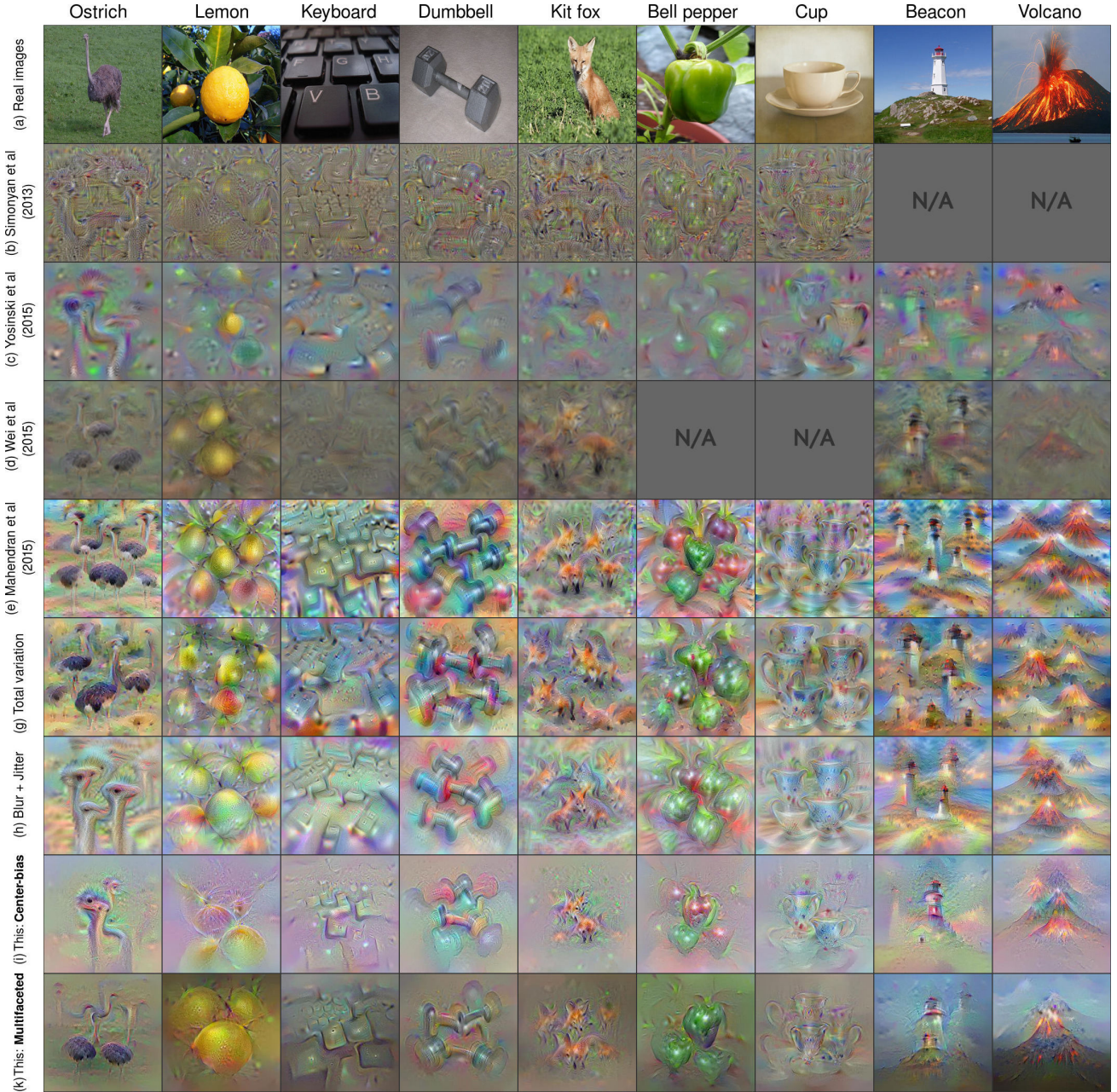
Figure S3. Comparing previous state-of-the-art activation maximization methods to the two new methods we propose in this paper: (k) *multifaceted visualization* (Sec. 3.3) and (i) *center-biased regularization* (Sec. 2.4). For a fair comparison, the categories were not cherry-picked to showcase our best images, but instead were selected based on the images available in previous papers (Simonyan et al., 2013; Yosinski et al., 2015; Wei et al., 2015). For (e), we reproduced the algorithm of Mahendran et al. (2015), which applies both jitter and TV regularizers, although we combined them in a slightly different way (see text in S2.1). Interestingly, we found that the visualizations produced by TV alone (g) are just as good as TV+Jitter (e), which are both better than previous methods (b-d). Overall, while it is a subjective judgement and readers can decide for themselves, we believe that multifaceted feature visualization produces more recognizable images with more natural colors and more realistic global structure.

Figure S4. Four facets each for example class output neurons (fc8) produced by multifaceted feature visualization. These images are hand picked to showcase the multifaceted nature of neurons across a variety of classes from natural to man-made subjects. The reconstructions are produced by applying Algorithm 1 with $k = 10$ on the ImageNet 2012 validation set. Best viewed electronically with zoom.

Figure S5. Visualizing the different facets of a neuron that detects bell peppers. Example, diverse facets include a single, red bell pepper on a white background (1), multiple red peppers (5), green peppers either on the plant (4), a cutting board (6), or against a dark background (10), and yellow peppers (8). Images from the training set from the "bell pepper" class (center) are geometrically projected into two dimensions by t-SNE and clustered by k-means (see text). The two columns are synthetic images generated by multifaceted feature visualization for the "bell pepper" class neuron for each of the 10 numbered facets. Best viewed electronically with zoom. This figure is an expanded version of Fig. 2 from the main text.

Figure S6. Visualizing the different facets of a neuron that detects images in the "fishing reel" class. Example, diverse facets include reels on backgrounds that are white (2), dark (3), ocean blue (7) or forest green (8), reels placed next to fish laying on grass (4), people fishing at sea (5), and a specific type of reel with holes in it (9). Each reconstruction is a facet visualization for a cluster of images in the "fishing reel" class. The image components are as described in Fig. S5, except next to each facet visualization we include the four images in each facet closest to the center of that facet cluster. Best viewed electronically with zoom.
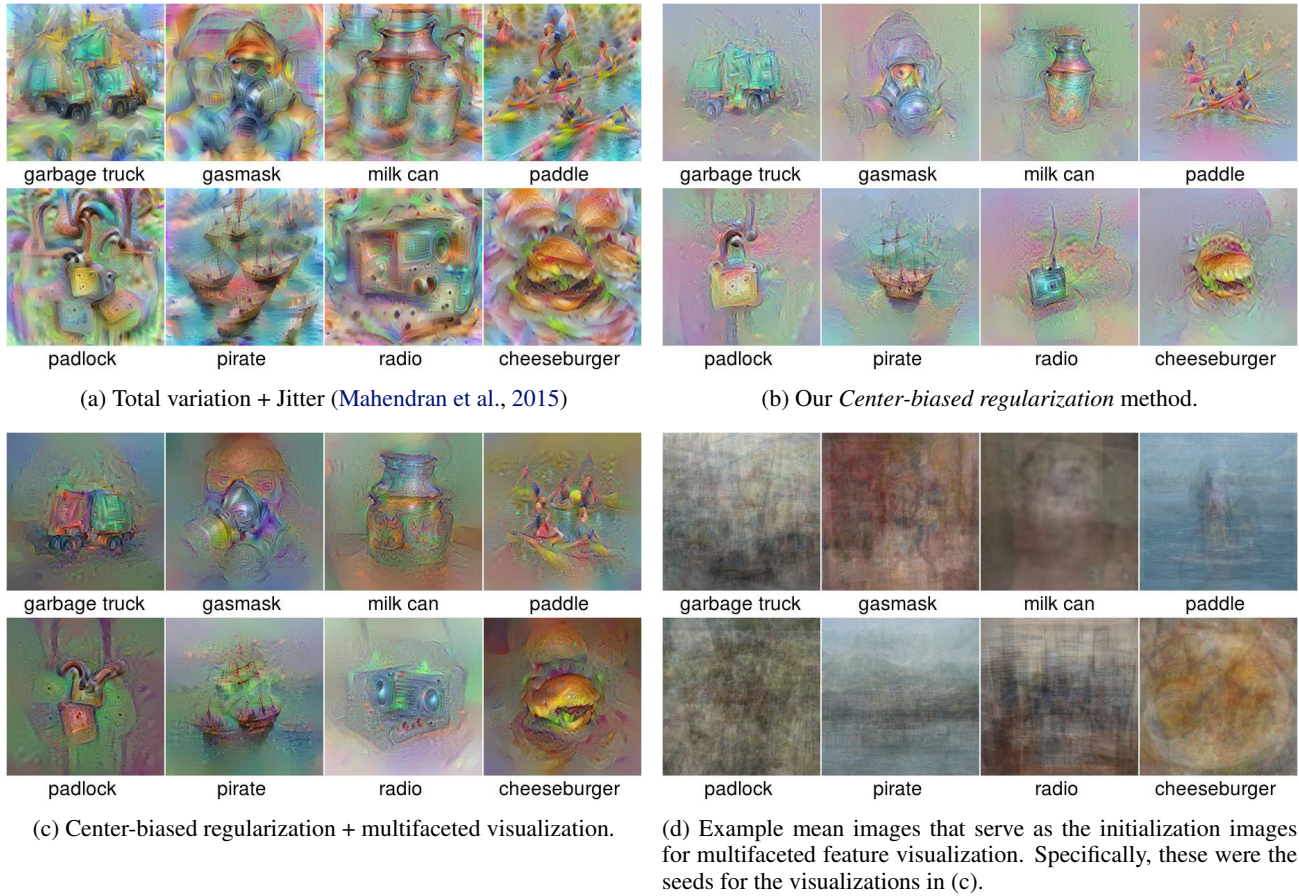
garbage truck   gasmask   milk can   paddle

padlock   pirate   radio   cheeseburger

(a) Total variation + Jitter (Mahendran et al., 2015)

garbage truck   gasmask   milk can   paddle

padlock   pirate   radio   cheeseburger

(b) Our *Center-biased regularization* method.

garbage truck   gasmask   milk can   paddle

padlock   pirate   radio   cheeseburger

(c) Center-biased regularization + multifaceted visualization.

garbage truck   gasmask   milk can   paddle

padlock   pirate   radio   cheeseburger

(d) Example mean images that serve as the initialization images for multifaceted feature visualization. Specifically, these were the seeds for the visualizations in (c).

Figure S8. Center-biased regularization (b) produces images that have fewer repeated fragments, thus reflecting better the concepts learned by DNNs. The realism of the colors of these images are improved when combined with multifaceted visualization (c).

S3. What are the hidden units in fully connected layers for?

We reported in the main text (Sec. 3.2) that neurons in hidden, fully connected layers often seem to be an amalgam of very different, abstract concepts. For example, one facet reconstruction of a neuron in fc6 looks like a combination of a scuba diver and a turtle (Fig. 6, leftmost fc6 neuron). Here, we document a series of visualization experiments to further shed light on the inner-workings of the fc6 and fc7 neurons visualized in Fig. 6. Note that the reconstructions in Fig. 6 were produced by running Algorithm 1 with $k = 10$ clusters on the top 1000 *validation set* images that most highly activate each hidden neuron.

To understand what a neuron is for, we begin by looking at the image clusters among the top 1000 validation set images that most highly activate a given neuron (Fig. S9), similar to the approach of visualizing the top 9 images that activate a neuron from Zeiler et al. (2014). We observe that many fc6 and fc7 neurons fire for various types of images. For example, the fc6 unit that resembles a combination of a scuba diver and a turtle (Fig. S9c, left neuron) fires for turtles and scuba divers, but also for sharks, bells, human faces and even trucks. Even when multifaceted feature visualization is seeded with a mean image of very different concepts, such as human faces or automobiles, optimization for this neuron consistently converges to very similar images that resemble a turtle with goggles (Fig. S9c, left). Note that unlike fc6 and fc7 neurons, the same algorithm produces very unique facet reconstructions for fc8 neurons (Fig. S4).

For the fc8 layer, the "restaurant" unit responds to far views of a room in day & night (Fig. S9a, top 2 rows), and close-up views of food plates (Fig. S9a, bottom 2 rows). As we expect from MFV and see for convolutional layers and the last (fc8) layer, it seems that the final reconstructions closely reflect the content of the real images in the clusters. It seems that the images in each cluster effectively pull optimization toward the respective facet.

In some cases, however, the unit respond to an odd assortment of different types of images. For the "ostrich" class, we observe that this neuron fires highly for many non-ostrich images such as leopard and lizard (Fig. S9a, right panel). In fact, the same observation holds for many units: 1 out of the top 9 validation set images that highly activate an fc8 neuron often comes from a different class (given that each class has 50 images in the $50,000$-image validation set). Thus, the multifaceted visualization reconstructions do not always reconstruct the types of images averaged into the mean image that it is initialized with.

While informative, these cluster images could also be misleading because it is not clear whether a given neuron fires for very different concepts (Fig. S9c, photocopiers, trucks, dogs, etc.) or it just fires for certain common details in those images. Here, we visualize the features in the images that are responsible for the activation of a given neuron via Deconv (Zeiler et al., 2014) and Layer-wise Relevance Propagation (LRP) (Bach et al., 2015a) methods (Fig. S10). The Deconv results are produced via the Deep-Vis toolbox (Yosinski et al., 2015), and the LRP results are produced by the LRP toolbox (Bach et al., 2015b).

The results show that Deconv visualizations often contain a lot of noise (Fig. S10, bottom row), thus making it hard to identify the most important regions in an image. This observation agrees with a recent evaluation of Deconv and LRP methods (Samek et al., 2015). For this reason, we also visualize the images with LRP algorithm. The LRP heatmaps for the fc6 "scuba diver and turtle" unit shows that this neuron fires indeed for underwater things like: scuba divers, sharks and turtles (Fig. S10a, top row in LRP panel). However, the same unit also fires for a variety of other concepts: turtles in black backgrounds, a metal bell, human heads, and automobile wheels (Fig. S10a, LRP panel). Noticeably, the most important (yellow highlighted) feature in all four reconstructions is the circular pattern that looks like goggles. We thus hypothesize that this neuron might indeed be an amalgam of different concepts; however, the main facet of this neuron can be described as a "turtle with goggles".

Our conclusion remains the same for many other fc6 and fc7 units (see also Fig. S10b) that: these neurons might indeed be an amalgam of different concepts; however, each neuron often has a main facet that is commonly reflected in every of their multifaceted visualizations: from layer fc6: "turtles with goggles" and "yellow dog" (Fig. S9c); and from layer fc7: "an arch" and "brownish-furred bird" (Fig. S9b). Another mutually exclusive hypothesis for why the reconstructions are not easily interpretable is that these neurons are part of highly distributed representations. Overall, the evidence in this section reveals how little we know about the precise function of hidden neurons in fully connected layers, and much work is required to make progress on this interesting question.
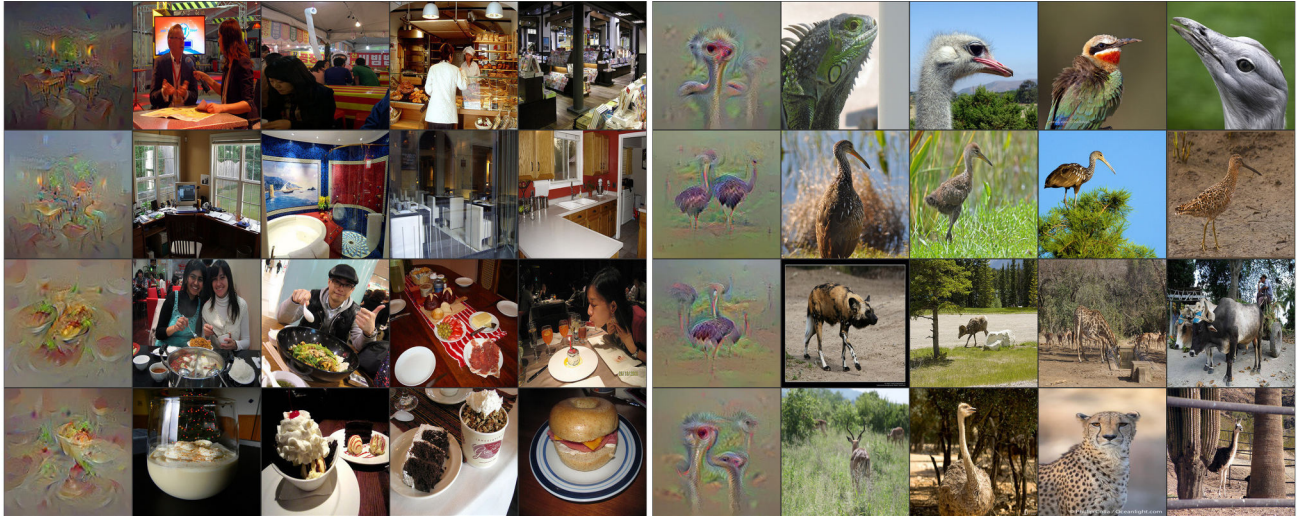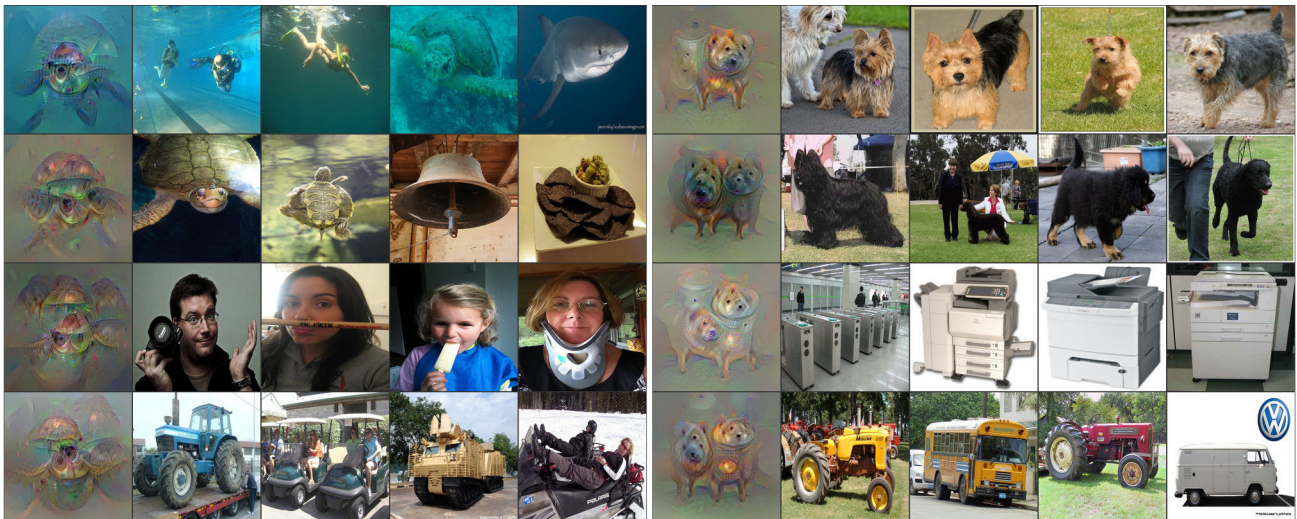
(a) fc8 units: "restaurant" (left) and "ostrich" (right).

(b) fc7 units indexed 159 (left) and 466 (right).

(c) fc6 units indexed 22 (left) and 461 (right).

Figure S9. To further shed light on the facet reconstructions of fc6 and fc7 neurons discussed in the main text (Fig. 6), we investigate two neurons from each of those layers. For each neuron, we show 4 different facets (one per row). For each facet, the leftmost image is the multifaceted visualization, and next to it are 4 random images from the 15 validation set images that are used for computing the mean image for each cluster. Unlike fc8 units, a hidden fc6 or fc7 unit tends to respond to a variety of different concepts (e.g. from animals to automobiles), and all of their facet reconstructions often converge to similar images. Note that these real images are among the top 1000 validation set examples that highly activate the neuron in question, and are grouped into clusters by their layer codes via t-SNE and k-means algorithms (see Sec. 3).

(a) fc6 unit that resembles "a scuba diver and a turtle".
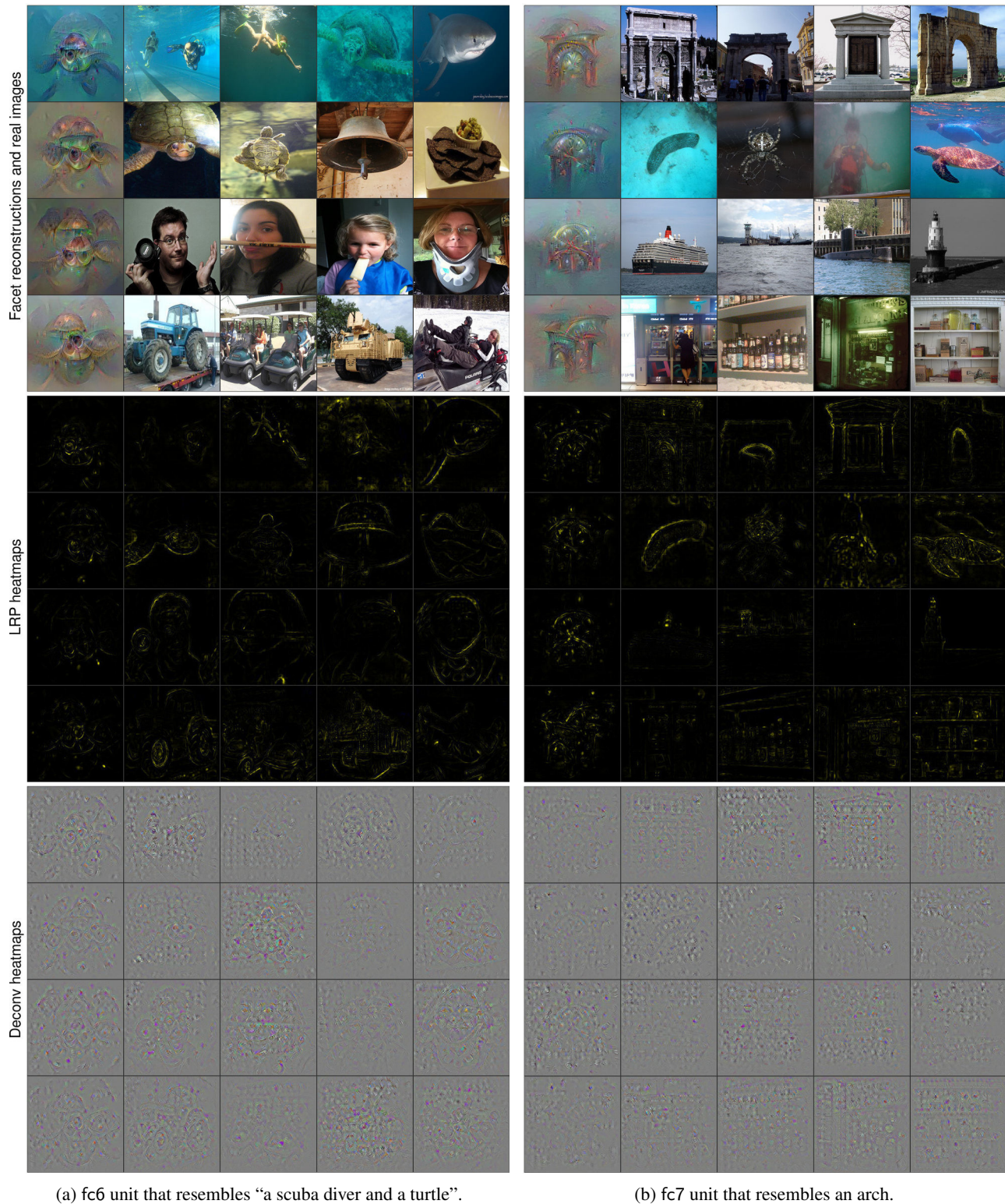
(b) fc7 unit that resembles an arch.

Figure S10. Visualizing input patterns that are responsible for the activations of an fc6 neuron and an fc7 neuron via Deconv (bottom panels) (Zeiler et al., 2014) and Layer-wise Relevance Propagation (middle panels) (Bach et al., 2015a). See Fig. S9 for the description of the top panels. Layer-wise Relevance Propagation heatmaps show that these two neurons indeed fire for a variety of objects, supporting the hypothesis that they are amalgams of different concepts, part of a distributed representation, or represent a non-obvious abstract concept. The Deconv visualizations show a lot of noise, making it hard to conclusively identify the most important regions in each image.