AMERICAN
Scientist

COMPUTING SCIENCE

Computer Vision and Computer Hallucinations

A peek inside an artificial neural network reveals some pretty freaky images.

Brian Hayes

http://People have an amazing knack for image recognition. We can riffle through a stack of pictures and almost instantly label each one: dog, birthday cake, bicycle, teapot. What we *can't* do is explain how we perform this feat. When you see a rose, certain neurons in your brain's visual cortex light up with activity; a tulip stimulates a different set of cells. What distinguishing features of the two flowers determine this response? Experiments that might answer such questions are hard to carry out in the living brain.

What about studying image recognition in an artificial brain? Computers have lately become quite good at classifying images—so good that expert human classifiers have to work hard to match their performance. Because these computer systems are products of human design, it seems we should be able to say exactly how they work. But no: It turns out computational vision systems are almost as inscrutable as biological ones. They are "deep neural networks," modeled on structures in the brain, and their expertise is not preprogrammed but rather learned from examples. What they "know" about images is stored in huge tables of numeric coefficients, which defy direct human comprehension.

In the past year or two, however, neural nets have begun to yield up a few fleeting glimpses of what's going on inside. One set of clues comes from images specially designed to fool the networks, much as optical illusions fool the biological eye and brain. Another approach runs the neural network in reverse; instead of giving it an image as input and asking for a concept as output, we specify a concept and the network generates a corresponding image. A related technique called *deep dreaming* burst on the scene last spring following a blog post from Google Research. Deep dreaming transforms and embellishes an image with motifs the network has learned to recognize. A mountaintop becomes a bird's beak, a button morphs into an eye, landscapes teem with turtle-dogs, fish-lizards, and other chimeric creatures. These fanciful, grotesque images have become an Internet sensation, but they can also serve as a mirror on the computational mind, however weirdly distorted.

enlarge image

Learning to See

The neurons of an artificial neural network are simple signal-processing units. Thousands or millions of them are arranged in layers, with signals flowing from one layer to the next.

A neural network for classifying images has an input layer at the bottom with one neuron for each pixel (or three neurons per pixel for color images.) At the top of the stack is a layer with one output neuron for each possible category of image. Between the input and output layers are "hidden" layers, where features that distinguish one class from another are somehow extracted and stored.

A newly constructed neural network is a blank slate; before it can recognize anything, it must be trained. An image is presented to the input layer, and the network proposes a label. If the choice is incorrect, an error signal propagates backward through the layers, reducing the activation of the wrongly chosen output neuron. The training process does not alter the wiring diagram of the network or the internal operations of the individual neurons. Instead, it adjusts the weight, or strength, of the connections between one neuron and the next. The discovery of an efficient "backpropagation" algorithm, which quickly identifies the weights that most need adjusting, was the key to making neural networks a practical tool.

Early neural networks had just one hidden layer, because deeper networks were too difficult to train. In the past 10 years this problem has been overcome by a combination of algorithmic innovation, faster hardware, and larger training sets. Networks with more than a dozen layers are now commonplace.

Some networks are fully connected: Every neuron in a layer receives input from every neuron in the layer below. The new image-recognition networks are built on a different plan. In most of the layers each neuron receives inputs from only a small region of the layer below—perhaps a 3×3 or 5×5 square. All of these patches share the same set of weights, and so they detect the same motifs, regardless of position in the image plane. The result of applying such position-independent filters is known as *convolution*, and image-processing systems built in this way are called *convolutional neural networks*, or *convnets*.

The convnet architecture creates a natural hierarchy of image structures. In the lower layers of the network each neuron sees a neighborhood of only a few pixels, but as information propagates upward it diffuses over wider areas. Thus small-scale features (eyes, nose, mouth) later become elements of a coherent whole (a face).

An annual contest called the ImageNet Large Scale Visual Recognition Challenge has become a benchmark for progress in computer vision. Contestants are given a training set of 1.2 million images sorted into 1,000 categories. Then the trained programs must classify another 100,000 images, trying to match the labels suggested by human viewers. Some of the categories are fairly broad (restaurant, barn), others much more specific (Welsh springer spaniel, steel arch bridge).

For the past three years the contest has been dominated by convnets. The 2014 winner was a system called GoogLeNet, developed by Christian Szegedy of Google and eight colleagues. The network is a 22-layer convnet with some 60 million parameters to be adjusted during training.

Seeing in Reverse

When a convnet learns to recognize a Welsh springer spaniel, what exactly has it learned? If a person performs the same task, we say that he or she has acquired a concept, or mental model, of what the dog breed looks like. Perhaps the same kind of model is encoded in the connection weights of GoogLeNet, but where should you look for it among those 60 million parameters?

One promising trick for sifting through the network's knowledge is to reverse the layer-to-layer flow of information. Among the groups exploring this idea are Andrea Vedaldi and Andrew Zisserman of the University of Oxford and their colleagues. Given a specific target neuron in the upper layers of the network, they ask what input image would maximize the target neuron's level of activation. A variation of the backpropagation algorithm can answer this question,

producing an image that in some sense embodies the network's vision of a flower or an automobile. (You might try the same exercise for yourself. When you summon to mind a category such as *measuring cup*, what images flash before your eyes?)

The reversal process can never be complete and unambiguous. Classification is a many-to-one mapping, which means the inverse mapping is one-to-many. Each class concept represents a potentially infinite collection of input images. Moreover, the network does not retain all of the pixels for *any* of these images, and so it cannot show us representative examples. As members of the Oxford group write, "the network captures just a sketch of the objects." All we can hope to recover is a murky and incomplete collage of features that the convnet found to be useful in classification. The dalmatian image has black and white spots, and the lemon image includes globular yellow objects, but many other details are missing or indecipherable.

Learning from Failure

Quite a lot of what's known about human cognitive abilities comes from studies of mental malfunctions, including the effects of injury and disease as well as more mundane events such as verbal errors and misinterpreted images. Two intriguing recent results apply this idea to image recognition in convnets.

A group led by Szegedy (the developer of GoogLeNet) harnessed an optimization algorithm to find "adversarial" images, specially crafted to fool a convnet classifier. Start with an image that the network correctly recognizes as a school bus, change a few pixels—changes so slight they are imperceptible to the human eye—and the network now assigns the image to another class.

Ahn Nguyen of the University of Wyoming, with Jason Yosinski and Jeff Clune, has performed a complementary experiment. They generated images that look to the human observer like pure noise, yet the network recognizes them with high confidence as a cheetah or a centipede.

These findings raise questions about the reliability and robustness of neural network methods, but those concerns should not be overblown. It is *not* the case that any small random change to an image is likely to mislead the classifier. As a matter of fact, convnets perform well even with heavy doses of random noise. The adversarial examples are so rare they will almost never be encountered by chance, yet their existence indicates that the network's training leaves "wormholes" where two distant regions of the image space are brought together.

"We Need to Go Deeper"

In June of this year an article posted on the *Google Research Blog* suddenly brought the mysteries of deep neural networks to the attention of a much wider audience. The post was accompanied by a gallery of outlandish but strangely engaging images that attracted interest not just from the computer vision community but also from artists, cognitive scientists, and the press and public. This new genre of graphic works was given the name *inceptionism*, alluding to a line in the science fiction film *Inception*: "We need to go deeper." A follow-up blog post introduced the term *deep dream*, which has caught on.

The algorithm behind the deep dream images was devised by Alexander Mordvintsev, a Google software engineer in Zurich. In the blog posts he was joined by two coauthors: Mike Tyka, a biochemist, artist, and Google software engineer in Seattle; and Christopher Olah of Toronto, a software engineering intern at Google.

Here's a recipe for deep dreaming. Start by choosing a source image and a target layer within the neural network. Present the image to the network's input layer, and allow the recognition process to proceed normally until it reaches the target layer. Then, starting at the target layer, apply the backpropagation algorithm that corrects errors during the training process. However, instead of adjusting connection weights to improve the accuracy of the network's response, adjust the source image to increase the amplitude of the response in the target layer. This forward-backward cycle is then repeated a number of times, and at intervals the image is resampled to increase the number of pixels.

As the iterations continue, ghostly patterns emerge from the image, faintly at first and then more distinctly. A dark smudge becomes a dog's nose, a wrinkled bit of cloth turns into a spider web, lighthouses and windmills sprout from the empty blue sky. The process is self-reinforcing. A neural network has within it a huge jumble of image elements drawn from the training set, many of which can be matched to random fragments of the source image. The network acts a bit like Hamlet feigning madness, when he looks at a cloud and sees first a camel, then a weasel, then a whale.

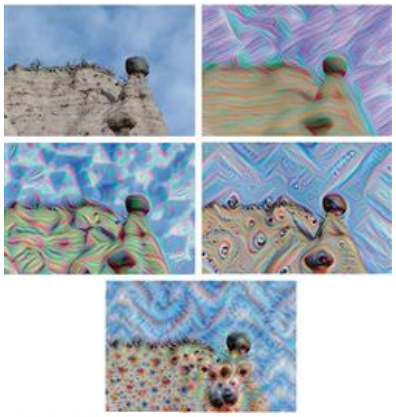
In an e-mail exchange I asked Mordvintsev, Tyka, and Olah how they came to invent their technique. I was surprised to learn that the original goal was solving a routine graphics problem: preventing loss of detail when enlarging an image. "We expected that maximizing the magnitude of current internal activations of the [convnet] on random patches of a slightly blurry image would add some of the missing details. Turned out it did."

A few weeks after their first blog post, Mordvintsev, Tyka, and Olah published their deep dream program, making it free for anyone to download and run. Others immediately began experimenting with the algorithms, and several websites now offer deep dreaming as a service. One company has packaged the code with a point-and-click interface for $15 (but it's not as versatile as the original).

The deep dream program itself is only about 100 lines of code, written in the Python programming language, but it relies on several other large frameworks and libraries, including a few that must be compiled from source code. If all goes well, installing the software takes a few hours. It did *not* go well for me on my first try, or my second. I finally succeeded by starting fresh with a blank disk drive.

Dream and Hallucination

"Dreaming," in my view, is not quite the right metaphor for this process. Dreams are what the mind conjures when the perceptual apparatus is shut down; here the visual system is hyperactive, and what it generates are hallucinations. In these images we witness a neural network struggling to make sense of the world. The training process has implanted expectations about how pieces of reality should fit together, and the network fills in the blanks accordingly. Photographs and other "natural" images—all those that might conceivably represent a three-dimensional scene on planet Earth—form a minute subset of all possible arrays of colored pixels. The network can only construct images consistent with this very special sample.



+ enlarge image

Many aspects of the images suggest a focus on purely local transformations. Faces, whether human or animal, generally have the proper complement of eyes, nose, and mouth, but the face may well be mounted on the wrong kind of body. Also, neural networks apparently can't count. Dogs are not limited to just four legs, or just one head. Yet there are also some global constraints that seem to be enforced throughout the image frame. However many legs an animal has, they all reach the ground. Objects of all kinds stand upright and rest upon a surface. The system can even create such a surface if necessary, turning a vertical wall into a "ground plane" seen in perspective. In some cases there's a rough sense of scale consistent with the perspective view: Big dog down front, tiny building on the horizon.

The most flamboyant dream images come from layers near the middle of the convnet stack, but the results from lower layers are also interesting, both aesthetically and for what they reveal about perceptual mechanisms. In the mammalian visual cortex some of the earliest stages of processing detect edges in various orientations, gradients, and other simple high-contrast forms such as center-surround patterns. It's fascinating to see that similar motifs turn up in the early layers of a convolutional neural network. And they were not put there by the programmer; they emerged from the network's own geometric analysis of the training set.

One could dismiss the deep dream technique as an overengineered contrivance for making funny-looking pictures, like an Instagram filter run amok. And indeed the fad may fade away as quickly as it came. So far, the methodology is documented only in source code and blog posts; if there is more scholarly work under way, it has not yet been published. Will anything of substance ever come out of this line of inquiry?

I don't know, but I have some questions I would like to see answered. In particular, why are certain kinds of content so heavily overrepresented in the dream images? The abundance of canines may reflect biases in the ImageNet database (120 of the 1,000 categories are dog breeds). Birds, spiders, ornate buildings, lanterns, and gazebos are also frequent, and eyes are everywhere. But where are the cats? All of these images were downloaded from the Web, which is supposed to be full of cats!

I would also like to know which geometric elements in the substrate image are most likely to be embellished. I thought I might approach this question by looking at the program's action on simple textures, such as a photograph of beach pebbles. It turns out that such planar patterns don't evoke much; the network seems to need 3D structure to stimulate the creative urge.

The freaky menagerie of deep dream images is both entertaining and distracting. I think it important to keep in mind that the underlying technology was designed not to generate these weird images but to recognize and classify ordinary ones. Furthermore, the program does that job quite well. The two-headed dogs and the sky spiders are evidently part of that process. The task now is to understand why.

Bibliography

- Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing,* vol. 25.
- Mahendran, Aravindh, and Andrea Vedaldi. 2014. Understanding deep image representations by inverting them. http://arxiv.org/1412.0035
- Mordvintsev, Alexander, Michael Tyka, and Christopher Olah. 2015. Inceptionism: Going deeper into neural networks, *Google Research Blog*, http://googleresearch.blogspot.ch/2015/06/inceptionism-going-deeper-into-neural.html. See also DeepDream—a code example for visualizing neural networks, *Google Research Blog*, http://googleresearch.blogspot.com/2015/07/deepdream-code-example-for-visualizing.html; and Deepdream code repository, https://github.com/google/deepdream.
- Nguyen, Ahn, Jason Yosinski, and Jeff Clune. 2015. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Computer Vision and Pattern Recognition 2015*.
- Russakovsky, Olga, et al. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, DOI:10.1007/s11263-015-0816-y.
- Simonyan, Karen, Andrea Vedaldi, and Andrew Zisserman. 2014. Deep inside convolutional networks: Visualising image classification models and saliency maps. http://arxiv.org/1312.6034.
- Szegedy, Christian, et al. 2014. Intriguing properties of neural networks. http://arxiv.org/1312.6199.
- Yosinski, Jason, et al. 2015. Understanding neural networks through deep visualization. 31st International Conference on Machine Learning. http://www.evolvingai.org/files/2015_Yosinski_ICML.pdf.

A note to my readers

It has been my privilege to write the *Computing Science* column since 1993. This is my 125th column, and it will be my last.

I thank my patient editors. I thank the many scientists and mathematicians who have generously shared their work, and guided mine. And I thank the readers of *American Scientist*—by far the most thoughtful and responsive audience I have ever had.

To answer some questions that often go unspoken: I have not been fired, and I am not retiring. On my agenda is learning more math, doing more computing, and writing all about it. If you would like to follow my further adventures, please stay tuned to http://bit-player.org.