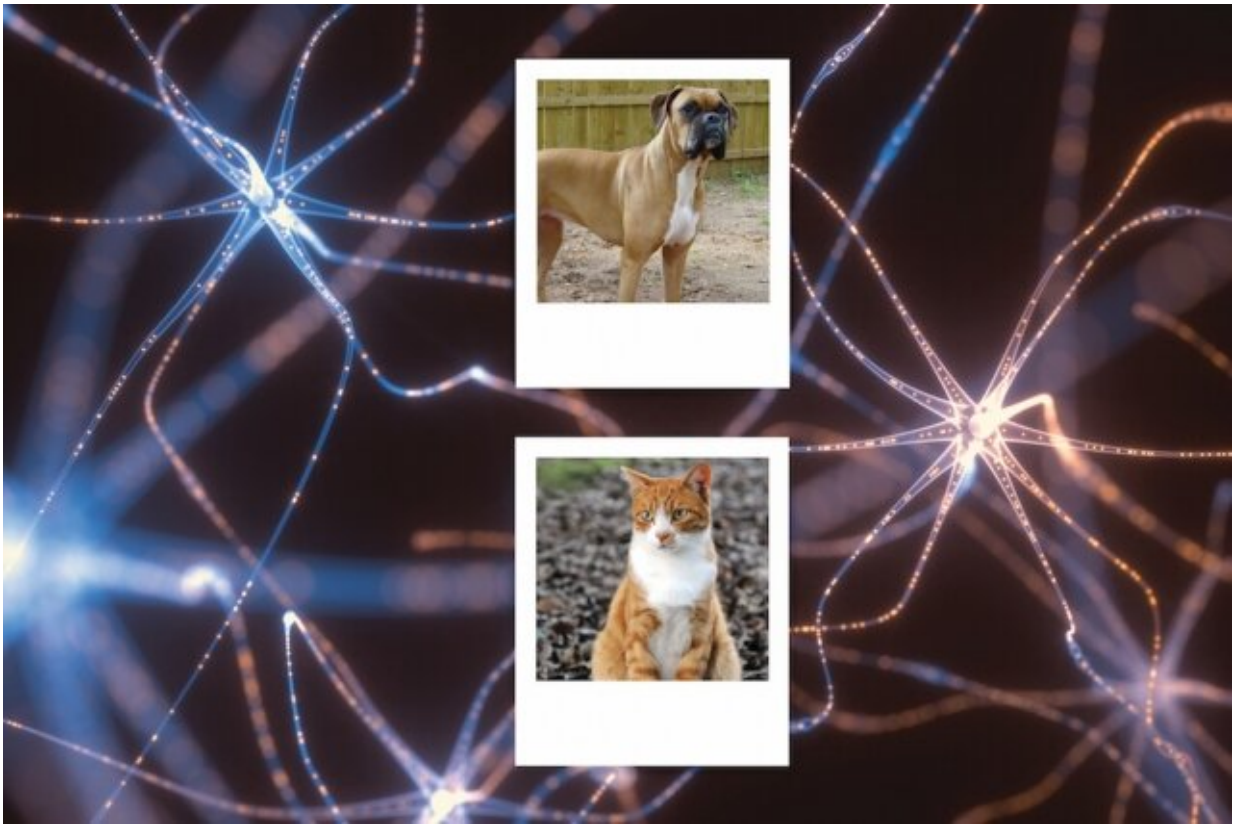
How to tell whether machine-learning systems are robust enough for the real world

May 10 2019, by Rob Matheson



Adversarial examples are slightly altered inputs that cause neural networks to make classification mistakes they normally wouldn't, such as classifying an image of a cat as a dog. Credit: MIT News Office

MIT researchers have devised a method for assessing how robust

machine-learning models known as neural networks are for various tasks, by detecting when the models make mistakes they shouldn't.

Convolutional neural networks (CNNs) are designed to process and classify images for computer vision and many other tasks. But slight modifications that are imperceptible to the human eye—say, a few darker pixels within an image—may cause a CNN to produce a drastically different classification. Such modifications are known as "adversarial examples." Studying the effects of adversarial examples on neural networks can help researchers determine how their models could be vulnerable to unexpected inputs in the real world.

For example, driverless cars can use CNNs to process visual input and produce an appropriate response. If the car approaches a stop sign, it would recognize the sign and stop. But a 2018 paper found that placing a certain black-and-white sticker on the stop sign could, in fact, fool a driverless car's CNN to misclassify the sign, which could potentially cause it to not stop at all.

However, there has been no way to fully evaluate a large neural network's resilience to adversarial examples for all test inputs. In a paper they are presenting this week at the International Conference on Learning Representations, the researchers describe a technique that, for any input, either finds an adversarial example or guarantees that all perturbed inputs—that still appear similar to the original—are correctly classified. In doing so, it gives a measurement of the network's robustness for a particular task.

Similar evaluation techniques do exist but have not been able to scale up to more complex neural networks. Compared to those methods, the researchers' technique runs three orders of magnitude faster and can scale to more complex CNNs.

The researchers evaluated the robustness of a CNN designed to classify images in the MNIST dataset of handwritten digits, which comprises 60,000 training images and 10,000 test images. The researchers found around 4 percent of test inputs can be perturbed slightly to generate adversarial examples that would lead the model to make an incorrect classification.

"Adversarial examples fool a neural network into making mistakes that a human wouldn't," says first author Vincent Tjeng, a graduate student in the Computer Science and Artificial Intelligence Laboratory (CSAIL). "For a given input, we want to determine whether it is possible to introduce small perturbations that would cause a neural network to produce a drastically different output than it usually would. In that way, we can evaluate how robust different neural networks are, finding at least one adversarial example similar to the input or guaranteeing that none exist for that input."

Joining Tjeng on the paper are CSAIL graduate student Kai Xiao and Russ Tedrake, a CSAIL researcher and a professor in the Department of Electrical Engineering and Computer Science (EECS).

CNNs process images through many computational layers containing units called neurons. For CNNs that classify images, the final layer consists of one neuron for each category. The CNN classifies an image based on the neuron with the highest output value. Consider a CNN designed to classify images into two categories: "cat" or "dog." If it processes an image of a cat, the value for the "cat" classification neuron should be higher. An adversarial example occurs when a tiny modification to that image causes the "dog" classification neuron's value to be higher.

The researchers' technique checks all possible modifications to each pixel of the image. Basically, if the CNN assigns the correct

classification ("cat") to each modified image, no adversarial examples exist for that image.

Behind the technique is a modified version of "mixed-integer programming," an optimization method where some of the variables are restricted to be integers. Essentially, mixed-integer programming is used to find a maximum of some objective function, given certain constraints on the variables, and can be designed to scale efficiently to evaluating the robustness of complex neural networks.

The researchers set the limits allowing every pixel in each input image to be brightened or darkened by up to some set value. Given the limits, the modified image will still look remarkably similar to the original input image, meaning the CNN shouldn't be fooled. Mixed-integer programming is used to find the smallest possible modification to the pixels that could potentially cause a misclassification.

The idea is that tweaking the pixels could cause the value of an incorrect classification to rise. If cat image was fed in to the pet-classifying CNN, for instance, the algorithm would keep perturbing the pixels to see if it can raise the value for the neuron corresponding to "dog" to be higher than that for "cat."

If the algorithm succeeds, it has found at least one adversarial example for the input image. The algorithm can continue tweaking pixels to find the minimum modification that was needed to cause that misclassification. The larger the minimum modification—called the "minimum adversarial distortion"—the more resistant the network is to adversarial examples. If, however, the correct classifying neuron fires for all different combinations of modified pixels, then the algorithm can guarantee that the image has no adversarial example.

"Given one input image, we want to know if we can modify it in a way

that it triggers an incorrect classification," Tjeng says. "If we can't, then we have a guarantee that we searched across the whole space of allowable modifications, and found that there is no perturbed version of the original image that is misclassified."

In the end, this generates a percentage for how many input images have at least one adversarial example, and guarantees the remainder don't have any adversarial examples. In the real world, CNNs have many neurons and will train on massive datasets with dozens of different classifications, so the technique's scalability is critical, Tjeng says.

"Across different networks designed for different tasks, it's important for CNNs to be robust against adversarial examples," he says. "The larger the fraction of test samples where we can prove that no adversarial example exists, the better the network should perform when exposed to perturbed inputs."

"Provable bounds on robustness are important as almost all [traditional] defense mechanisms could be broken again," says Matthias Hein, a professor of mathematics and computer science at Saarland University, who was not involved in the study but has tried the technique. "We used the exact verification framework to show that our networks are indeed robust … [and] made it also possible to verify them compared to normal training."

This story is republished courtesy of MIT News (web.mit.edu/newsoffice/), a popular site that covers news about MIT research, innovation and teaching.

Provided by Massachusetts Institute of Technology

learning-robust-real-world.html