Researchers unveil tool to debug 'black box' deep learning algorithms

25 October 2017

A debugging tool developed by researchers at Columbia and Lehigh generates real-world test images meant to expose logic errors in deep neural networks. The darkened photo at right tricked one set of neurons into telling the car to turn into the guardrail. After catching the mistake, the tool retrains the network to fix the bug. Credit: Columbia Engineering

Computers can now beat humans at chess and Go, but it may be a while before people trust their driving. The danger of self-driving cars was highlighted last year when Tesla's autonomous car collided with a truck it mistook for a cloud, killing its passenger.

Self-driving cars depend on a form of machine learning called deep learning. Modeled after the human brain, layers of artificial neurons process and consolidate information, developing a set of rules to solve complex problems, from recognizing friends' faces online to translating email written in Chinese. The technology has achieved impressive feats of intelligence, but as more tasks become automated this way, concerns about safety, security, and ethics, are growing. Deep learning systems do not explain how they make their decisions, and that makes them hard to trust.

In a new approach to the problem, researchers at Columbia and Lehigh universities have come up with a way to automatically error-check the thousands to millions of neurons in a deep learning neural network. Their tool, DeepXplore, feeds confusing, real-world inputs into the network to expose rare instances of flawed reasoning by clusters of neurons. Researchers present it on Oct. 29 at ACM's Symposium on Operating Systems Principles in Shanghai.

"You can think of our testing process as reverse engineering the learning process to understand its logic," said co-developer Suman Jana, a computer scientist at Columbia Engineering and a member of the Data Science Institute. "This gives you some visibility into what the system is doing and where it's going wrong."

Debugging the neural networks in self-driving cars is an especially slow and tedious process, with no way of measuring how thoroughly logic within the network has been checked for errors. Manually-generated test images can be randomly fed into the network until one triggers a wrong decision, telling the car to veer into the guardrail, for example, instead of away. A faster technique, called adversarial testing, automatically generates test images it alters incrementally until one image tricks the system.

DeepXplore is able to find a wider variety of bugs than random or adversarial testing by using the network itself to generate test images likely to cause neuron clusters to make conflicting decisions. To simulate real-world conditions, photos are lightened and darkened, and made to mimic the effect of dust on a camera lens, or a person or object blocking the camera's view.

A photo of the road may be darkened just enough, for example, to cause one set of neurons to tell the car to turn left, and two other sets to tell it to go right. Inferring that the first set misclassified the photo, DeepXplore automatically retrains the network to recognize the darker image and fix the

bug.

Using optimization techniques, researchers have designed DeepXplore to trigger as many conflicting decisions with its test images as it can while maximizing the number of neurons activated.

DeepXplore also generates test images that mimic the effect of dust on a camera lens. The pixelated photo at right tricked one set of neurons into telling the car to turn into a building. After catching the mistake, the tool retrains the network to fix the bug. Credit: Columbia Engineering

Testing their software on 15 state-of-the-art neural networks, including Nvidia's Dave 2 network for self-driving cars, the researchers uncovered thousands of bugs missed by previous techniques. They report activating up to 100 percent of network neurons—30 percent more on average than either random or adversarial testing—and bringing overall accuracy up to 99 percent in some networks, a 3 percent improvement on average.

Still, a high level of assurance is needed before regulators and the public are ready to embrace robot cars and other safety-critical technology like autonomous air-traffic control systems. One limitation of DeepXplore is that it can't certify that a neural network is bug-free. That requires isolating and testing the exact rules the network has learned.

A new tool developed at Stanford University, called ReluPlex, uses the power of mathematical proofs to do this for small networks. Costly in computing time, but offering strong guarantees, this small-scale verification technique complements DeepXplore's full-scale testing approach, said ReluPlex co-developer Clark Barrett, a computer scientist at Stanford.

"Testing techniques use efficient and clever heuristics to find problems in a system, and it seems that the techniques in this paper are particularly good," he said. "However, a testing technique can never guarantee that all the bugs have been found, or similarly, if it can't find any bugs, that there are, in fact, no bugs."

DeepXplore has application beyond self-driving cars. It can find malware disguised as benign code in anti-virus software, and uncover discriminatory assumptions baked into predictive policing and criminal sentencing software.

"We plan to keep improving DeepXplore to open the black box and make machine learning systems more reliable and transparent," said co-developer Kexin Pei, a graduate student at Columbia. "As more decision-making is turned over to machines, we need to make sure we can test their logic so that outcomes are accurate and fair."

The team has made their open-source software public for other researchers to use, and launched a website to let people upload their own data to see how the testing process works. "We want to make it easy for researchers to be able to validate their machine learning systems," said co-developer Junfeng Yang, a computer scientist at Columbia Engineering and a member of the Data Science Institute. "Creating the next generation of programming and validation tools for this new computing paradigm will require a collaborative effort that will ultimately benefit society."

Adds co-developer Yinzhi Cao, a computer scientist at Lehigh: "Our ultimate goal is to be able to test a system, like self-driving cars, and tell the creators whether it is truly safe, and under what conditions."

More information: Kexin Pei et al, DeepXplore, *Proceedings of the 26th Symposium on Operating Systems Principles - SOSP '17* (2017). DOI: 10.1145/3132747.3132785