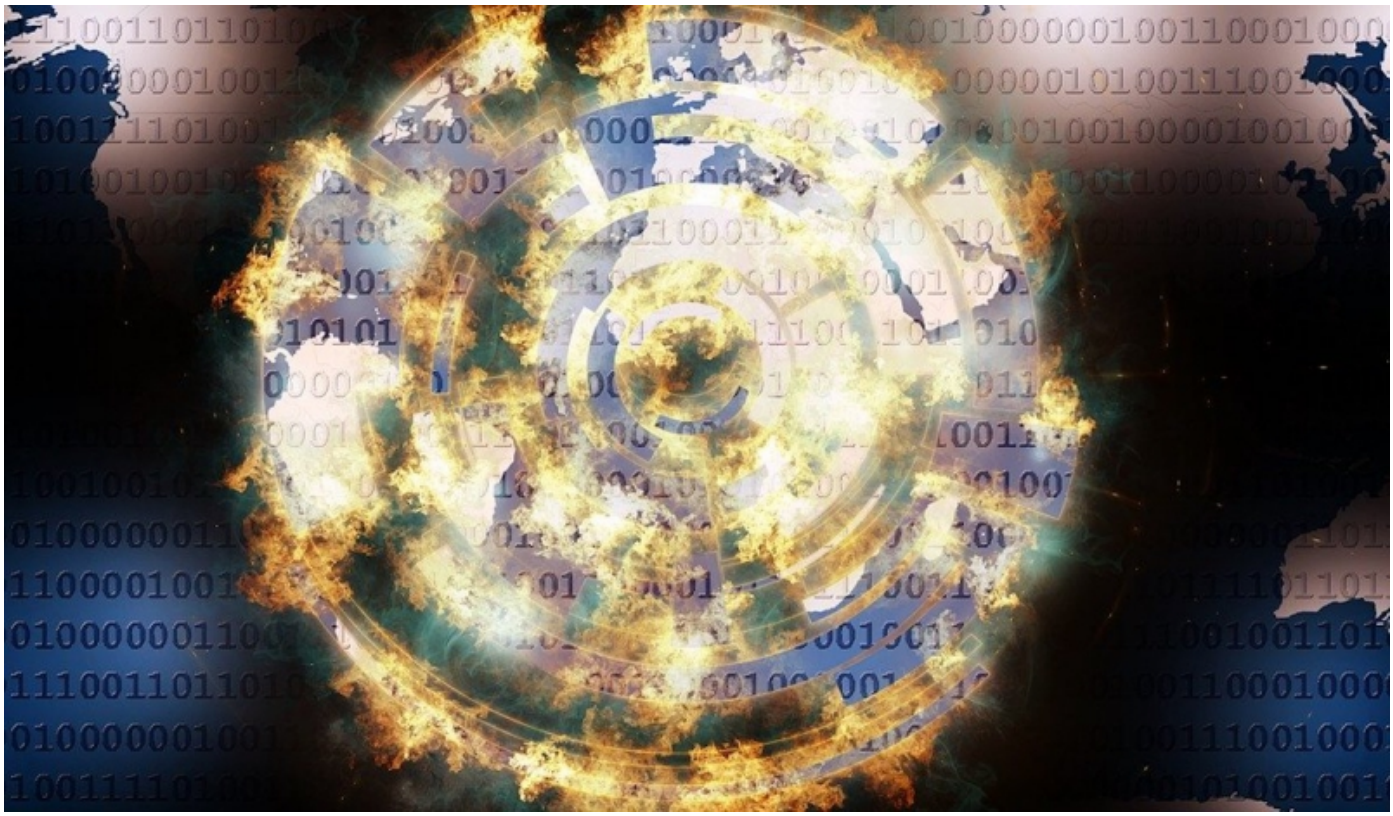
BOREALIS AI

Protecting networks with neural networks

May 29, 2017



While not as high-profile as some of its more recent brethren, the October 2015 attack on TalkTalk, a telecom company in the UK, still managed to send chills across the ether.

Using an SQL injection, cyber-attackers infiltrated customer data with the ease of a casual pickpocket lifting wallets on a crowded subway, accessing close to 157,000 files and costing the company nearly $700,000 (CAD) in fines alone. At the time, it was the highest fine ever issued by the Information Commissioner's Office (ICO) and set a new standard for corporate dread.

SQL – or **Structured Query Language** – is a process that more or less lets you interface with a database. In a standard database interaction, you would enter your

BOREALIS AI

However, someone could also enter a SQL command instead of a user name to *delete* all your information and that, to use the technical term, is *no bueno*. For example, there's a lady whose surname is "Null" and as you can probably imagine she has a really hard time booking flights.

The challenge

We explored the use of NLP for analyzing server logs. The goal was to automatically classify between benign and malicious logs by capturing the language properties of each class. While server code is not natural language, the hypothesis was that this data could also be processed in a similar manner.

What did we do?

We accessed over 50,000 server logs from security audits that were split roughly 50/50 between benign and malicious interactions, and fed these examples letter by letter into a neural network. In developing our model, one of the key decisions hinged upon choosing the right method of language processing. Standard NLP approaches might typically involve word embedding, but in this case, we knew it wouldn't be the best route, mainly because code isn't a standard language: there's non-standard punctuation, parentheses, and every letter, semi-colon, and asterisk counts. Therefore, it made sense to approach the problem from a **character-embedding level**.

Our algorithm was composed of two different types of neural networks – **convolutional** and **recurrent**. The convolutional network's job was to pick at the pattern in the text and reduce dimensionality. The recurrent network's job was to remember what it's seen, so it looked at the past history as it started processing the new information that helped it form context.

We also employed Batch Normalization and Dropout to improve accuracy and reduce co-dependency among the hidden states.

BOREALIS AI

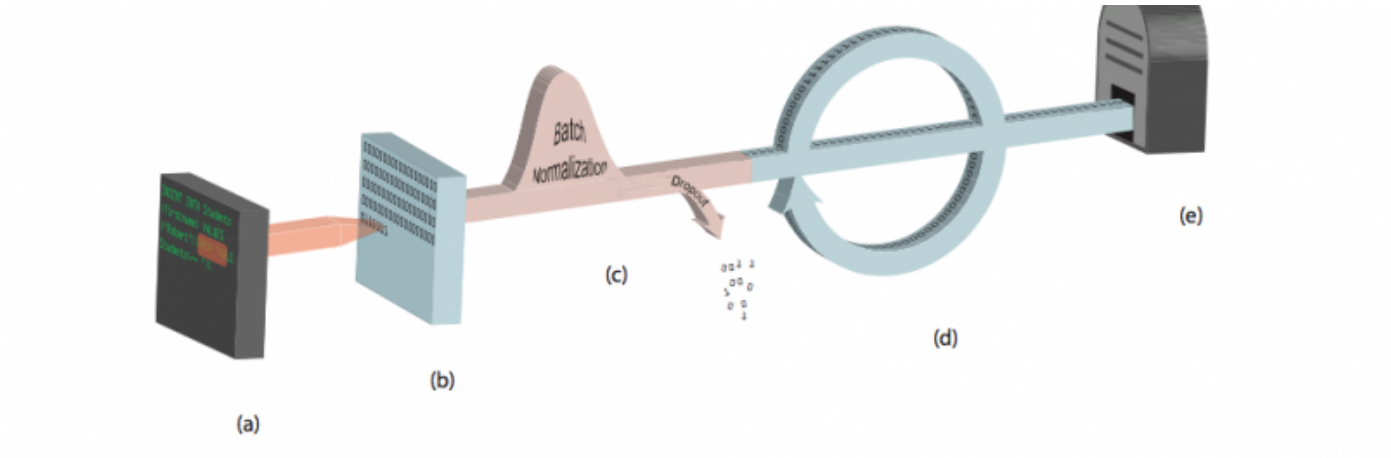


Figure 1: Illustration of the malicious code neural network. A SQL injection (a) is read into a convolutional layer (b) which picks up patterns in the text. These features are then regularized via batch normalization and dropout (c). Next they pass through the recurrent layer (d), which remembers the sequence of patterns, and generates new features. Finally these features go through the classification layer (e) which labels the SQL as malicious.

But the most exciting bit, at least according to the project's lead researcher and lead Irish person, **Cathal Smyth**, was the visualization technique we developed along the way. One of the biggest criticisms neural networks get is that they're basically a black box – we sometimes don't know why they make the decisions they make. While our algorithm gave us excellent results in terms of accuracy, precision, and recall, we still wanted to be able to eyeball the results and see if they made sense. For instance, when a hidden unit encountered text that it considered relevant, we wanted to be able to chart that activation.

BOREALIS AI

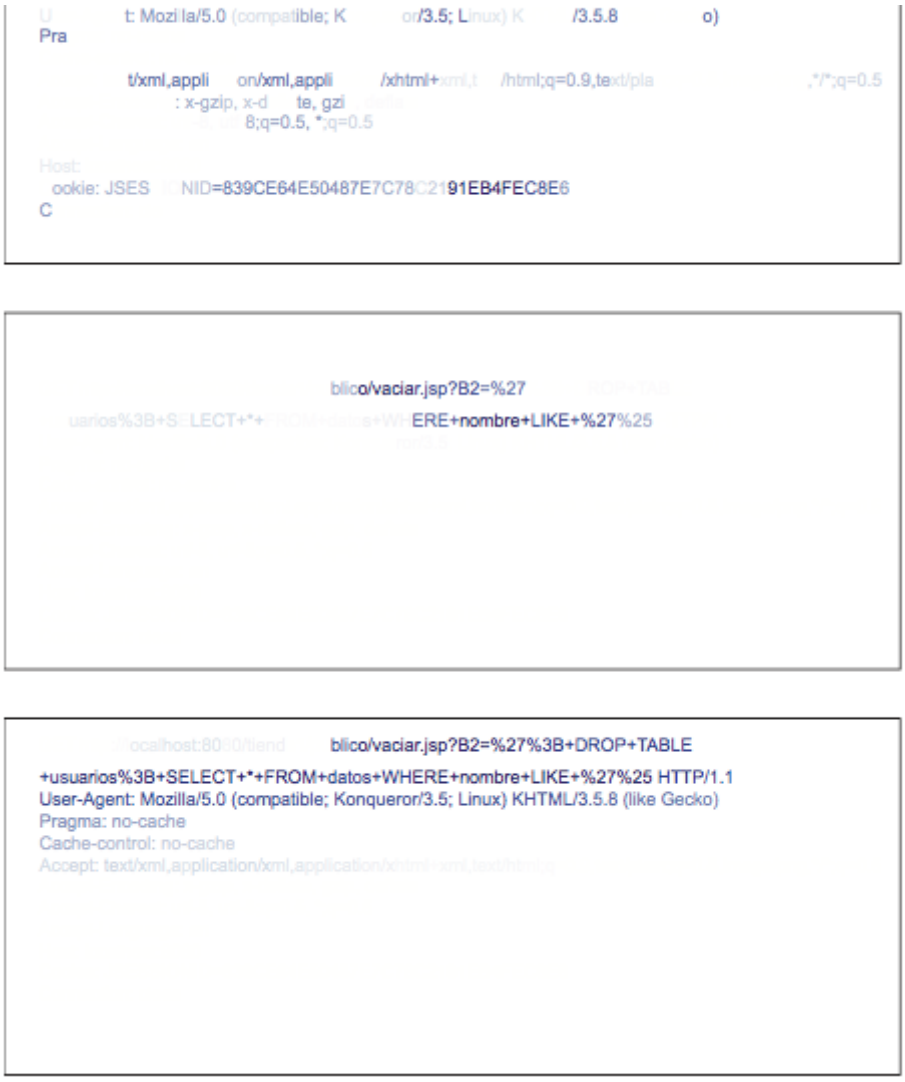


Figure 2: An example of a SQL injection, represented by hidden nodes from 3 different layers.

Take a look at the sweet three-layer cake of a diagram above. In the first image, you can see the hidden unit from the first layer is picking out letters that may almost seem random. By the second layer, it's much easier to start seeing the algorithm select relevant words. By the third and final layer, it has accurately identified the entire attack.

The results

After training the network, we managed to achieve accuracy levels of 99 per cent, or equivalent to a human expert, only orders of magnitude faster: Roughly 1700 samples per second versus a human expert average of one sample per second. In a standard rules-based approach, speed is variable and depends on the number of

BOREALIS AI

What happens next?

We're now seeing more algorithms that are able to generate text. We may be a long way off from that evolving into the generation of working code, but there will come a day when people will be able to whip up some malicious code on the fly and we have to be ready for that. When that happens, the complexity, scale, and frequency of attacks are going to snowball.

Our solution could be deployed in parallel to whatever enterprise cybersecurity is being used now to protect networks. Moreover, you can use those evil code generators to "pen -test" applications before launching them.

But the really exciting thing isn't how we taught a machine to become an expert; rather, how the machine taught a human to be an "expert" in cybersecurity. Cathal and Kory now claim that thanks to the algorithm, they are able to recognize numerous cyberattacks despite their mere *homo sapiens* status. Their weekend favourite is the double encoding attack, an approach hackers use to infiltrate security systems by url-encoding the payload, then doubly encoding it if security catches on. With the proposed representation learning solution, one can pick out the fractal-like pattern created in double encoding malicious attacks.

BOREALIS AI