INFERRING THE FUTURE

Probability theory is coming to the rescue of systems at risk of making overconfident mistakes. By **Chris Edwards**

One aspect of deep learning that surprised researchers in the technology's early days a decade ago, and remains puzzling even now, lies the ability of the models built during training to generalise.

A typical deep-learning model contains so many neurons, and with that an even larger number of trainable connection weights and other parameters, that conventional wisdom points to a strong risk of overfitting. AI experts initially expected the models to, more or less, encode the training data and be unable to handle images or sounds they had not seen before.

The reverse turned out to be the case. A curious result that, according to recent research may be a spinoff benefit from randomness that is applied during training to speed up the process.

On the other hand, deep-learning models suffer from an unfortunate tendency to be wrongly confident when asked to come up with answers during inferencing. Instead of overfitting, they overgeneralise and suffer from delivering woefully inaccurate results when the real-world data strays too far from the training set. If deep learning is to become a permanent fixture in robots and self-driving cars, it needs to be less sure of itself.

This is why research into uncertainty in machine learning and in other areas of data-driven computing is now coming to the fore.

Cambridge University professor and Uber chief scientist Zoubin Ghahramani argues probabilistic techniques should make it easier to build AI systems that act more rationally. His team has used techniques such as Bayesian inference and Gaussian processes to provide ways of generating results from deep-learning models that are presented alongside reasonable estimates of uncertainty. He also believes that application of probability theory can help control the complexity of learned models, something he calls the "Bayesian Occam's Razor".

The idea revolves around the use of Bayesian inference to determine how likely it is that each of a group of candidate models built during training would generate results that fit a randomly selected set of test data.

A further reason for moving to more probabilistic forms of AI is that these techniques show more promise where data is hard to get because they can extend the range of a model, albeit one that gets less confident as the inputs move away from the data in the training set. The big issue with deep learning is that IT eats data. It's one reason why the cloud-computing companies seem ever keen to soak up data from wherever they can get it.

Many problems simply do not have the huge quantities of information these systems need. In medicine, for example, doctors' records are far

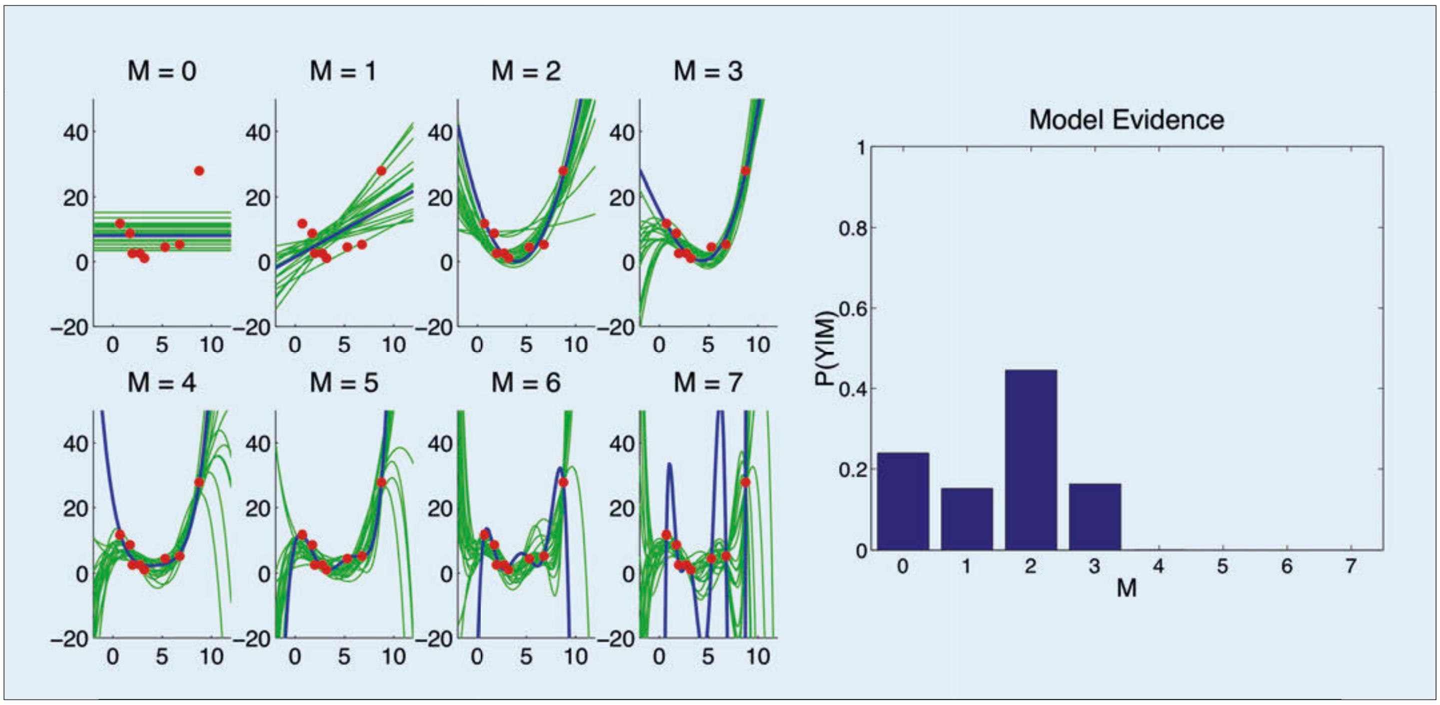

Figure 1: Bayesian Occam's Razor uses the probability of a particular model – in this case a linear function – fitting a set of points picked at random from the relevant data but which does not fit random data just as well. The one with the highest probability wins, which will typically be the model with the lowest reasonable complexity. (Source: Zoubin Ghahramani)

from comprehensive for individual patients and it does not make sense to merge data across a lot of different patients and attempt to learn from that set. The genetic and environment differences will simply blur the results.

More robust applications

It is not just in AI. This focus on probabilistic methods is spilling over into more conventional areas of programming and companies like Intel have started to push funding into the area in the hope of uncovering techniques that can drive the next wave of computing and embedded processing. Using probabilistic methods, some researchers believe they can build more robust applications by propagating information about uncertainty through each operation rather than treating each variable as being as precise as its datatype.

Embedded and control systems engineers arguably have more experience dealing with the false confidence of precision than their peers in cloud or desktop computing. Kalman and particle filters now form an important part of many control systems, both taking into account the role of uncertainty in measurements. The Kalman filter assumes the noise in data recorded by sensors follows a Gaussian distribution. Every time it updates a set of matrices that represent the system's state, it compares the new value of each input with a predicted state based on previous behaviour. The closer the sensor data matches the prediction, the lower the assumed error, which in turn feeds into the gain of the Kalman filter.

For example, if the data from a new GPS fix, for example, points to a larger than expected change in position, the gain for that reading is set to a low value. Accelerometer and gyroscope may be treated as more important and so avoid the problem found with early sports watches that made it look as though the runner was leaping around like Spiderman.

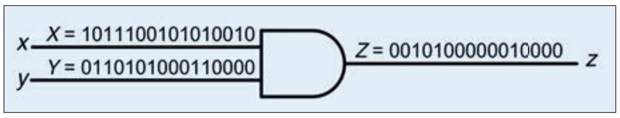

Figure 2: An AND gate as a stochastic multiplier. (Source: University of Michigan)

The Kalman filter employs the relatively simple method of weighted averages to create its estimates. The particle filter works better for non-linear systems and is another technique that applies aspects of Bayesian probability theory.

To make dealing with imprecision a more general-purpose tool, researchers are developing specialised languages, including Uber's own Pyro, as well as libraries for conventional programming languages that can associate quantities such as error bars with data elements.

One effort from Microsoft Research was Uncertain<T>, which was intended as a way of providing programmers with easy access to statistical functions.

A more recent example from Microsoft was a framework for uncertainty propagation, initially implemented for the Hadoop MapReduce environment. Although this first library was focused on server-based processing, the approach provides the opportunity for trade-offs with the processing performed by IoT and embedded systems. An embedded device might reduce the precision of its work in the expectation that later processing would reflect any uncertainty in the data it produces. For time-series data, the uncertainty of any single results is unlikely to matter much, just as long as downstream algorithms are not biased by bad samples. The servers need to do more work to handle the uncertainty using techniques such as uncertainty propagation but the IoT device gets a longer battery life in return.

The idea behind IoT devices being less accurate is that it could use a lot less power. At voltages that are close to the switching threshold of digital logic, it becomes extremely hard to determine whether calculations will be correct. However, if the algorithms are designed to tolerate bit-level mistakes, it becomes much easier to operate the device much close to that threshold and take advantage of the quadratic gains in power efficiency that come with lower voltages. The problem for more approximate computing architectures based on binary numbers is that it becomes extremely hard to identify how big the mistakes could be if a voltage sag means individual gates handling bits of different magnitude fail to complete before their clock period is up.

"This focus on probabilistic methods is spilling over into more conventional areas of programming."

One approach to efficient approximation is to change how data gets encoded. Rather than encode numbers in conventional binary representations, stochastic computing treats a 'packets of bits' as a probability of the variable being in a particular state. The more bits in the packets that are set to one the higher the probability of a variable being in a particular state. The shorter the packet, the more uncertain the value. As bit in the stream so do mistakes, such as bit flips.

The stochastic-computing representation is very similar to that produced by the 1bit modulators in sigma-delta A/D converters. Rather than applying a decimation filter to convert that into a binary number, stochastic computing simply works on the stream.

A big advantage of stochastic computing is that functions such as multiplication can be performed using just a single AND gate – albeit rather slowly as each bit is processed in turn in the simplest possible implementation. However, for systems based on harvested energy, such simple circuits may provide a way to eke more processing out of limited resources as long as back-end systems understand the limitations of the data they provide. And, in general, greater understanding of uncertainty could well stop systems from making much bigger mistakes.