Neural Networks: Crash Course Statistics #41

Crash Course: Statistics
https://youtube.com/watch?v=JBlm4wnjNMY
https://nerdfighteria.info/v/JBlm4wnjNMY

Hi, I'm Adriene Hill, and Welcome back to Crash Course Statistics.

In previous episodes we've talked about things like cars learning how to drive themselves...and apps that can recognize handwriting and turn it into printed text. A lot of these projects are done using a type of Machine Learning called a Neural Network.

The term Neural Network covers a bunch of different--but related--methods that can take in data and spit out useful outputs. Neural networks can output everything from the probability of someone getting a particularly nasty strain of MRSA on their next hospital stay, to new chapters of Harry Potter...seriously. They may even be behind some of the annoying Twitter bots that just seem to spout tweets that rile people up.

Today, we're going to take a look at the big picture of what neural networks are, and how they do all these things. INTRO In Crash Course Computer Science, we talked a little bit about what a neural network is. In the simplest sense, a neural network looks at data and tries to figure out the function--or set of calculations--that turns the input... variables...into the output.

That output could be a number, a probability, or even something a bit more complicated. Neural networks are analogous to robots that can learn to make things---like a toy car--not by following step by step instructions from humans, but by looking at a bunch of toy cars and figuring out for itself how to turn inputs (like metal and plastic) into outputs (the toy cars)! If we want to work with data instead of toy cars we can use a neural network to predict future salary based on a number of variables such as degree, field, age, years of experience, gender, number of promotions, and university.

We feed these variables to the neural network. These circles are called Nodes, and they just hold a value like degree or field. Eventually we want the Neural Network to output its prediction for future salary.

So we know there will be one output node at the end of our network that tells us what it predicts the salary will be . At this point, the Neural Network looks kinda like a regression, we have a bunch of inputs...our variables...which are combined in some way to create an output...our predicted value. But unlike most regressions, neural networks feed the weighted sum of age, degree, field, etc through something called an "activation function" which takes the value and transforms it before returning an output.

These activation functions improve the way many neural networks learn, and give them more flexibility to model complex relationships between input and output. One common activation function is called Rectified Linear Unit (ReLU) --which turns all negative values to 0, and leaves positive ones as they are. This makes these nodes act a little bit like neurons in your brain--hence the name neural network--which require a certain "threshold" of activation before they'll fire.

So a node with 0 doesn't fire, or contribute to the output at all. But one with a positive value will. This Neural Network currently has two layers--input and output.

But we can add layers between them. So now the inputs are indirectly connected to the output, through the middle layer of nodes. It's pretty clear what the input nodes are, since they're values we understand.

And the output node is a salary, so we get that too. But it can be harder to grasp exactly what the middle layers represent. You can think of all the calculations that happen between the input nodes and output nodes as something called "feature generation". "Feature" is just a fancy word for a variable that can be made up of a combination of other variables.

For example we could use your grades, attendance, and test scores to create a "Feature" called Academic Performance. Essentially the neural network is taking the variables we give it, and performing combinations and calculations to create new values, or "features". Then, it combines those "features" to create an output.

When we have large amounts of complex data, the neural network saves us a LOT of time by combining variables and figuring out which ones are important. Neural Networks allow us to make use of data that might seem too big and overwhelming for us to try to use on its own. They can find patterns that humans might never be able to see.

If a neural network has more than one layer, we say that we're using "Deep Learning", since there are many layers of nodes. Deep Learning has gained popularity in recent years. Neural networks and deep learning have been used extensively to do things like recognize handwritten numbers and simulate x-ray images so airport security can be trained to recognize items like drugs and guns.

There's a lot more math that goes into neural networks . But in short, they learn by figuring out what they got wrong, and then working backwards to determine what values and connections made the output incorrect. For example, if it predicted my salary and is $10,000 off, it will take that difference and figure out which parts of the neural network were influential in creating that $10,000 error.

It then tweaks them so that next time, it's not as wrong. You can see that in this neural network--sometimes called a Feed Forward Neural Network--all the nodes only feed into the next layer from input to output. Hence, they only Feed information Forward.

But it is possible to feed the output of a Neural Net back into the model as an input the next time you run it. In other words, nodes in one layer can be connected to each other, even themselves! These types of Neural Networks are called Recurrent Neural Networks.

We can use RNNs to learn patterns. For example, words! RNNs have been used to spell check text.

The Network can learn to take in a misspelled word like this... and correct it. Often we use this kind of network when we have sequential data-- like stock prices over time, or the words in a sentence. If you're trying to predict the words in a sentence, it matters a lot what the previous word was.

If the previous word was "A", that influences what the current word is. Usually the word "A" precedes a noun, or an adjective -- one that starts with an consonant. A Fox.

A Quick, Brown Fox. But it's unlikely to precede a verb. "A walked" wouldn't make sense. But the further you get through the sentence, the less influence the word "A" has.

Unlike Feed Forward Neural Networks, Recurrent Neural Networks " remember " the previous outputs. For example, if we used a Recurrent Neural Networks to generate a melody, we would give the network some information about our song framework, and we'd ask it for a note. Then we feed that note back into the model along with the information about our song framework and the network would generate the next note.

In order to make a melody that sounds good, the Recurrent Neural Network needs to "remember" what the previous notes were. Using the outputs as inputs allows us to do that. A popular type of Recurrent Neural Network called a Long Short-Term Memory Network has been used to generate all kinds of music.

It's even been used to write a few new Harry Potter chapters. ahem Here is one of those chapters from a Recurrent Neural Network trained by Max Deutsch… "The Malfoys!" said Hermione. Harry was watching him. He looked like Madame Maxime.

When she strode up the wrong staircase to visit himself. "I'm afraid I've definitely been suspended from power, no chanceâ€Š—â€Šindeed?" said Snape. He put his head back behind them and read groups as they crossed a corner and fluttered down onto their ink lamp, and picked up his spoon. The doorbell rang.

It was a lot cleaner down in London. So, J. K.

Rowling isn't out of a job yet. This excerpt doesn't make sense within the context of the Harry Potter universe, or really make sense at all. But it at least has the structure of a book chapter.

We can also use Neural Networks to look at another form of art: images. A lot of applications of image recognition use a type of Neural Network called a Convolutional Neural Network. Images are made up of a grid of pixels.

A very tiny grayscale image like this could be represented by a grid like this ...where each number represents how much black is in that pixel. 0 is complete black, 1 is complete white, and anything in between is a shade of gray. Color images are a little more complicated, since each pixel has a red, green, and blue value, but the idea is similar. In this case, a pixel is affected by all the pixels surrounding it.

It's not simple sequential data. So, convolutional neural networks look at "windows" of pixels instead of one pixel at a time. They apply a filter to these windows to create "features".

This step is called convolution. The filters that the network uses are just calculations that transform the pixels that are inside the window. The network uses the data to determine which windows and filters will be used.

Some filters might help detect edges in the image Others might recognize features like curves, horizontal lines, or even more complex objects like eyes, or faces. These features make it so we can take an image...which has a huge number of pixels...and make a smaller number of features. This process is called pooling.

In the end, the network will use the features generated by convolution and pooling to give us some kind of output, like a decision about whether or not an image contains a stop sign, or a human face. Snapchat, for example, has used variations of convolutional neural networks in their app. And these networks are used extensively in all kinds of image recognition.

If you hate those captchas that ask you to click on each image that has a stop sign, you could use a convolutional neural network to fill them out for you. And the next time you're in another country, you can use Google's Translate app which uses these networks to help translate the text from signs or menus into your language. One thing that limits our use of neural networks of all kinds is a lack of data.

The more complex these networks are, the more data they need to perform well. But some neural networks can be trained to generate data. These are called Generative Adversarial Networks (GANs).

They use sets of existing data to try to learn how to create new data. These networks are kinda like two neural networks...disguised as one...by wearing a trench coat. We'll illustrate how they work, with an analogy.

Let's say you're a counterfeiter who's trying to make fake $100 bills. You examine a few $100 bills create a fake and then try to use it at your local convenience store. If the bill is rejected, you politely ask the cashier what made them realize the bill was fake.

And they're happy to help. They tell you, you take this information back to your counterfeiting lab, and make a new, better fake $100 bill. You repeat this process over and over--hopefully the cashiers don't start to recognize you...and eventually, you should have a passable fake bill. (Assuming you aren't already in jail.) However, since the cashiers are seeing so many fake bills, they get better at recognizing them as time goes on.

In our analogy, you are the generator. Your job is to make fake input...in this case $100 bills that are good enough to "trick" the convenience store. The cashier is the discriminator since her job is to learn to discriminate between real and fake $100 bills.

Essentially you have two neural networks battling it out to create better and better outputs. The generator is trying to get better and better at making data that can trick the discriminator. And the discriminator is trying to learn how to best discriminate between fake and real data.

These networks have been used to create new anime characters , make new Van Gogh-like art, and create new skate decks Neural Networks of all kinds help us deal with the big, sometimes messy data that we have in real life. They help detect patterns in data that humans can't see. And often, they're way better than we are at figuring out how to turn input variables into accurate outputs.

While you and I don't have time to look through Terabytes of data, a Neural Network does. With neural networks we can make use of data that may have been too overwhelming without them. We've talked about Neural Networks being applied to a lot of different things, like image and face recognition.

That type of technology could one day be utilized in drones on search-and-rescue missions. Natural Language Processing...often using neural networks...is the reason we have Alexa. Also, hey Google, hey Siri, hey Cortana, hey Bixbi.

I think I got everybody. As data gets bigger and bigger, neural networks will likely become a more common tool for making the most of our data, and understanding how they work allows us to think of really creative ways to use them. Hopefully, less annoying ways.

Thanks for watching, I'll see you next time.