Neural network for dummies

ML Enthusiast | Mechatronics Engineering StudentToday, with open source machine learning software libraries such as TensorFlow, Keras or PyTorch we can create neural networks, even with high structural complexity, with only a few lines of code. Having said that, the mathematics behind neural networks is still a mystery to some of us and having the mathematical knowledge behind neural networks and deep learning can help us understand what's going on inside the neural network. It is also useful in selecting architecture, fine-tuning deep learning models, hyperparameter tuning and optimization. Introduction I ignored the understanding of mathematics behind neural networks and deep learning for a long time because I did not have a good knowledge of algebra or differential calculus. A few days ago I decided to start from scratch and derive methodology and mathematics behind neural networks and deep learning, to know how and why they work. I also decided to write this article which would be useful for people like me who find it difficult to understand these concepts. PerceptronsPerceptrons – Invented by Frank Rosenblatt in 1957, are the simplest neural networks that consist of n number of inputs, only one neuron and one output where n is the number of functions of our data set. The process of transmitting data over a neural network is known as forward dissemination and the forward dissemination in Perceptron is explained in the next three steps. Step 1: For each input, multiply the input value xi by the weight of wi and the sum of all multiplied values. Weights — represent the strength of the connection between neurons and decide what effect a given input will have on the neuron output. If the weight w啦 has a higher value than the mass w啦, then input x啦 will have a greater effect on the output than w啦. Vectors of rows of inputs and weights are x = [x啦, x啦, ... , xn] and w =[w啦, w啦, ... , wn] and their dot product is givenHence, the sum is equal to the dot product vectors x and wStep 2: Add bias b to the sum of multiplied values and call it z. Bias – Also know how to offset it is necessary in most cases to move the entire activation function left or right to generate the desired output values. Step 3: Pass the value from on the nonline activation function. Activation functions — are used to introduce non-linearity into the output of neurons, without which the neural network will be a linear function only. In addition, they have a significant impact on the learning speed of the neural network. Perceptrons have binary step functions as their activation functions. However, we will use Sigmoid – we also know how logistical function as our activation function.where, σ sigmoid indicates the activation function and output we have after forward prorogation is known as the estimated value of the啦. Learning algorithm The learningucation algorithm consists of two parts – Backpropagation and Optimization.Backpropagation : Backpropagation, short for reverse promotion of errors, refers to the algorithm for calculating the gradient of function loss with respect to weights. However, this term is often used to refer to the entire learning algorithm. The backpropagation performed in Perceptron is explained in the following two steps. Step 1: To know the estimate of how far we are from our desired solution is used by the loss function. In general, a medium square error is selected as a loss of function for regression problems and cross entropy for classification problems. Consider the regression problem and its function loss is a mean error on the second, which squares the difference between the actual (yi) and the estimated value (啦i). The loss function is calculated for the entire training dataset and their average is called Cost Function C.Step 2: In order to find the best weights and bias for our Perceptron, we need to know how the cost function changes in relation to weights and bias. This is done with the help of transitions (rate of change) - how one quantity changes in relation to another quantity. In our case, we need to find the inclination of the cost function with respect to weight and bias. Calculate the inclination of cost function C with respect to wi weight using partial derivation. Since the cost function is not directly related to the weight of the wi, let's use the string rule. Now we need to find the following three gradientsLet to start with the gradient cost function (C) with respect to the estimated value (啦)Let y = [y啦 , y啦 , ... yn] and 啦 =[啦啦, 啦啦啦 , ... 啦n] are vectors of rows of actual and predicted values. Therefore, the above equation makes it easier for asNow let's find the gradient of the estimated value with respect to the z. It's going to be a little lengthy. Gradient from considering the weight of wi jeFore we get what Bias? — Bias is theoretically considered to be a constant value input of 1. Therefore,Optimization: Optimization is the selection of the best element from some set of available alternatives, which in our case is the selection of the best scales and perceptron bias. Let's choose the descent gradient as our optimization algorithm that changes weights and bias, proportional to the negative gradient cost function with respect to the corresponding weight or bias. Learning rate (α) is a hyperparameter that is used to control how much weight and bias change. The scales and bias are updated as follows and the backporpagation and gradient descent are repeated until convergence. Conclusion I hope you found this article and understood the mathematics behind neural networks and deep learning. I explained the work of one neuron in this article, but these basic concepts are applicable to all kinds of neural networks with some modifications. If you have any questions or if you found an error, please let me know in the comments. Become Hackolyte Level into your reading game by joining Hacker Noon Now! This picture was created by a real person. I guess. Illustration: Sam WoolleyArtificial Intelligence (AI) seems poised to run most of the world these days: he's detecting skin cancer, looking for hate speech on Facebook, and even marking possible lies in police reports in Spain. But AI are not all run by mega-corporations and governments; You can download some algorithms and play with them yourself, with often hilarious results. There's a faux Coachella poster full of fake band names, created feeding a bunch of real band names into neural networks and asking her to come up with some of his own. There are recipes created in a similar way, where barbecue beef calls for 1 beer - cut into cubes. And then there's my favorite, Janelle Shane's AI-generated colors (tag yourself, I'm Dorkwood). People aren't nearly as creative as we think they are. Craft brewers, for example, have run out of fun names Read more However, they have all been made with neural networks, a type of AI modeled after the network-like nature of our own brains. You train a neural network by giving it input: recipes, for example. The network strengthens some connections between its neurons (imitation brain cells) more than others as it learns. The idea is that it's figuring out the rules of how entry works: which letters tend to follow others, for example. Once the network is trained, you can ask it to generate its own output, or give it partial input and ask it to fill out the rest. But the computer does not really understand the rules of, say, making recipes. He knows that beer can be an ingredient, and that things can be cut into cubes, but no one has ever said that beer is not one of those things. Outputs that look almost correct but don't understand some of the basic rule are often the most hilarious. I was happy to just watch these antics from afar, while Shane mentioned on Twitter that the high school coding class had produced better ice cream names than her. And I thought if the kids could do it, I could do it. How to train your first neural NetI began with the same toolkit Shane uses for ice cream flavors: a python module called textgenrnn, max woolf from Buzzfeed. You will need basic command line knowledge to work with it, but it works on any system (Mac, Linux, Windows) where you have installed a programming language/interpreter python. Before you can train your own neural networks, you will need some input to begin with. Central class began with a list of thousands of ice cream flavors, for example. Whatever you choose, you'll want at least a few hundred examples; thousands would be better. Maybe you would like to download all your tweets, and ask the network to let you generate some new tweets. Or check out Wikipedia's list of lists for ideas. Whatever you choose, insert it into a text file with one item per line. It may take some creative copy-and-paste or spreadsheet work, or if you have an old hand on coding, you can write some nasty pearl scripts to munge the data into submission. I'm an ugly perl script kind of girl, but when I ended up wanting Lifehacker subtitles for one of my data sets, I just asked our analysis team for a great list of headlines and they emailed me exactly what I needed. Asking nicely is an underrated coding skill. (If you'd like to feed Lifehacker headlines into your own neural network, here's this list. There are about 10,000 of them.) Create a folder for the new project and type two scripts. First, one called train.py: from textgenrnn import textgenrnnt = textgenrnn()t.train_from_file('input.txt', num_epochs =5)This script gets a neural clean reading of your input and think about what its rules must be. The script has a few things you can edit: t =textgenrnn() is fine the first time you run the script, but if you would like to return to it later, type the name of the .hdf5 file that miraculously appeared in the folder when you started it. In this case, the line should look like this: t =textgenrnn('textgenrnn_weights.hdf5') 'input.txt' is the file name with one caption/recipe/tweet/etc per line. num_epochs is how many times you want to process the file. The neural network gets the better the longer you let you study, so start with 2 or 5 to see how long it takes and then go up from there. It takes a while to train the network. If you use scripts on a laptop, one epoch may take 10 or 15 minutes (larger datasets will take longer). If you have access to a muscular desktop, maybe your or a friend's gaming computer, it will go faster. If you have a large data set, you may want to ask for a few dozen or even hundreds of eras, and let it run overnight. Next, write another script called spit_out_stuff.py (you have the opportunity to give these better names than I did): from textgenrnn import textgenrnnt = textgenrnn ('textgenrnn_weights.hdf5')t.generate (20, temperature = 0.5) This is the fun part! The script above will give you 20 fun new things to look at. Important parts of this last line are: The number of things you want to generate: here, 20.The temperature that is like a creativity dial. At 0.1, you get a very basic output that's probably even more boring than what you've fed in at 1.0, the output gets so creative that often what comes out isn't even real words. You can go higher 1.0, and with Dare. When you run a training script, you'll notice that it displays sample output at different temperatures, so you can use it to guide how many epochs you're running and what temperature you want to use to generate the final output. Not every idea your neural network comes up with will be comedy gold. You will have to choose the best ones yourself. Here are some of the better Lifehacker headlines that my AI came up with: The best way to make a Baby LaptopHow to survive backspace drinking gameThe best way to buy a Job InterviewHow to get the best bonfire of your life with This handy GraphicsHow to make your own Podcast BarHow to get a new iPhone X If you ArduinoHow to clean up your own measurements in a museumHow to start with your stories and anxietyThe best way how to make your own ink from WinterHow to your relationship with the imaginary conceptU the best way to make the perfect cup of wine with raspberry piThe best way to eat toilet strawberryHow to get a better job on your vacation The best way to eat stubborn JarI got these by playing with the temperature and number of training eras, and every time I saw something I liked I copied into the text file of my favorites. I also experimented with the word-by-word version of the algorithm; the scripts above use the default character model. My final list of headlines includes results from both. If you're curious about some of the rejects, here's what I get with temperature 0.1: The best way to stay streaming stop more alternatives to your phoneThe best way to stream the best performance when you don't need to know about the worldThe best way to stay started started your common ways to stop anyoneHow to get the best way to see the best popular posts The best way to stay started started your common ways to stop anyoneHow to get the best way to see the best popular posts The best way to stay started your common ways to stop anyoneHow to get the best way to see the best popular posts The best way to stay started how to stay started to your PhoneA when I crank it up to 1.5 (dangerously creative): The remains of the day: How to Ad-Finger UnsubualRenew Qakeuage on Travel History, Ovenchime, or Contreiting PassfledThe Risk-Idelecady't Two-Copyns, Focus ZoomitasIfo Went Vape Lyrics Battery Oro crediblacy Supremee BuldsweoapottiesDIY Barbecue now Edt My Hises Uniti spread your wordsClean, human help is needed. Become a BuddyEven AI although neural networks can learn from data sets, they don't really understand what's going on. That's why some of the best results come from partnerships between people and machines. I know it's a tool I use, says Janelle Shane, but it's hard not to think of it as-'come on a small neural network, you can do it' and 'Oh, it was smart' or 'You're still a confused, poor little thing.'If you want to make the most of your relationship, you have to lead your friend's AI. Sometimes it can be so good to guess the rules of the dataset that it just recreates the same things. you have fed – version of AI You will need to check that its funny output is really original. Botnik studios pairs people with machines training predictive-text keyboards. Imagine picking up your friend's phone and writing messages using predictive text on your keyboard. Eventually, you'd write your own message, but in a style that sounds like your friend's style. In the same way, you can train a botnik keyboard with any data source you want, and then type with the words supplied by the keyboard. This is where this amazing advice column duel came from: two botnik keyboards trained on Savage Love and Dear Abby. If you'd rather work against, rather than with, your algorithmic buddy, look at how Janelle Shane joked about neural networks that initially seemed good at recognizing sheep grazing in the meadow. She photoshopped out the sheep, and realized AI was just looking for white balls in the grass. If she turns sheep's orange, AI thought they were flowers. So she asked her Twitter followers for sheep in unusual places and found that AI thinks the sheep in the car must be a dog, the goats in the tree must be birds, and the sheep in the kitchen must be a cat. Serious AI can have similar problems, and playing with algorithms for fun can help us understand why they are so error-prone. For example, one early skin-cancer-detection AI accidentally learned the wrong rules for telling the difference between cancerous and benign skin lesions. When the doctor finds a large lesion, they often photograph it next to the ruler to show the size. AI randomly taught itself that it is easy to recognize cancerous tumors: just look at the rulers. Another lesson we can learn is that the output of the algorithm is only as good as the data you feed. ProPublica found that one algorithm used in sentencing was tougher on black defendants than white supremacists. It does not consider race as a factor, but its entry led to a belief, incorrectly, that the crimes and background common to black defendants were stronger predictors of repeat crimes than crimes and backgrounds associated with white defendants. This computer had no idea about the concept of race, but if your input data reflects bias, the computer may end up maintaining this bias. It's best that we understand this limitation of algorithms and not assume that because they are not human, they must be impartial. (Good luck with hate speech AI, Facebook!) Mix datasetsYuchan't stop on a single dataset; You can mix up two of them and see what results. (I combined product lists from Goop and Infowars stores, for example. Slightly NSFW.) You can also train a classification algorithm. Shane says she already had a list of metal bands and a list of my little pony names, so she trained a classifier to make up the difference. (Pinky Doom: 99 percent metal.) Once you have a classifier trained, you can feed into it and get reading. Benedict Cumberbatch: 96 per cent metal. You can also feed anything you like into a trained textgenrnn network. When you specify how many items you want and how much temperature (creativity) you want the network to use, you can also provide a prefix. Then try to come up with words that should follow this prefix. After I trained Lifehacker subtitles, I asked AI to give me subtitles starting with 3 Folders Happy Hour. It responded with some wonderful fictional cocktails (again these are my picks from a longer list): 3 Ingredients Happy Hour: Herb Stressful Upgrade3 Ingredient Happy Hour: Cake Strawbreak3 Ingredient Happy Hour: Darkled Pot3 Ingredients Happy Hour: Pizza and Drinks For They Are Merchant Wings3 Ingredients Happy Hour: Ferrent Pot3 Component Happy Hour: Throat to Refreshing3 Ingredient Happy Hour: Best Bar Order3 Component Happy Hour: Leftover Party Controci3 Component Happy Hour: Summer Rum Cutting3 Folder Happy Hour: Best Coconati3 Folder Happy Hour: Beautiful Shicline3 Folder Happy Hour: Cheekey CandyYou should not be surprised if you see these in a future Lifehacker post; Claire Lowery, our food and drink editor, says she wants to try to do some of them. But instead of waiting for her expert recipes, I decided to feed these into the neural network as well. I gathered some cocktail recipes from Chris Lowder's cocktail guide and WikiBooks cocktail dictionary, and arranged them so that each cocktail took up one line of text file, titled cocktail as the first few words. That said, I could choose a cocktail name and ask my cocktail-trained neural network to make the recipe that follows. Here are some results: Best Coconati - oz. Benedictine e. 1 dash Aromatic b. &lt;1 oz. Cranberry d... 5 oz. Lemon c... 75 oz. Iteloun d. 2 Dashes Juponged Slipes i. Stir/Strain/Coupe/No garnishThe Cheekey Candy i. 1 oz. Blendey Sherry b. 1.5 oz. Fresh pineappbed d. Lonstine Brandy Bowl De there at large Jamaic c. 2 Dashes Pineappl d. 1 Dash Aromatic Bitters e. 1 dash Aromatic gin ii. 1 oz. Vodka II. . 5 oz. Aged Rum c. 2 Dashes Angostura Bitters i. Stir / Strain / Nick &amp; Nora Glass / Ice / 1The Ferrent Pot - - 1.25 oz. Green Chartreuse 1.5 oz. London Dry Gin b... 75 oz. Fill whiskey b. Orange half of whiskyYou can ask for anything, of course: Beth Skwarecki - 1 oz. Blended Scotch (Juice) Water b. 1 oz. Egg white in large rocks of glass with dets 1934 or makes Babbino Lifehacker c. 14 Vodka Martini i. . 75 oz. Campari i. Shake/Gentle Strain/Coupe/Lemon twist Entry details were just a few hundred cocktail recipes, so I had to turn the temperature up so up to get something interesting. And at high temperature (1.0, in this case), sometimes you get words that aren't really words. Good luck finding any Lonstine Brandy or Sherry in the store-but if you do, my pet AI will be very happy. Happy.