|  |  |
| --- | --- |
| Cornell Note Taking | |
| **Title (hyperlinked**): https://www.youtube.com/watch?v=njKP3FqW3Sk | |
| **Unit or Concept**: Why Deep Learning? | |
| **Citation Info**: Deep Learning and Sigmoid function | |
| **Key Ideas or Questions** | **Notes** |
| Why Deep Learning and how do computers recognize a mouth, eye etc.? | Traditional machine learning algorithms typically try to define as set of rules or features in the data and these are usually hand engineered and because their hand engineered, they often tend to be brittle in practice.  Let's take a concrete example if you want to perform facial detection how might you go about doing that?  First you might say to classify a face the first thing and classify or recognize if I see a mouth in the image the eyes ears and nose if I see all of those things then maybe I can say that there's a face in that image but then the question is but how do I recognize each of those sub things like how do I recognize an eye?  How do I recognize a mouth and then you have to decompose that into to recognize a mouth?  I maybe must recognize these pairs of lines-oriented lines in a certain direction certain orientation and then it keeps getting more complicated and each of these steps. |
| Define the set of features for deep learning to identify and image | To define a set of features that you are looking for in the image now and the key idea of deep learning is that you will need to learn these features just from raw data.  Take a bunch of images of faces and then the deep learning algorithm is going to develop some hierarchical representation of first detecting lines and edges in the image using these lines and edges to detect corners and eyes and mid-level features like eyes noses mouths ears then composing these together to detect higher-level features like maybe jaw lines side of the face etc. which then can be used to detect the final face structure and actually the fundamental building blocks of deep learning.  These building have existed for decades and they are under underlying algorithms for training these models have also existed for many years. |
| Big Data and connection to logarithms | We're living in a the age of big data and these algorithms are hungry for a huge amounts of data to succeed secondly these algorithms are massively parallel which means that they can benefit tremendously from modern GPU architectures and hardware acceleration that simply did not exist when these algorithms were developed. |
| Open source toolboxes -tensor flow | Due to open source toolboxes like tensor flow deploying these models has become extremely streamlined. |
| Perceptron | Fundamental building block of a neural network which is a single neuron or what is also called a perceptron the idea of a perceptron or a single neuron. |
| All data in real life is nonlinear! | Forward propagation of information through a neuron is defined as a set of inputs to that neuron as x1 through XM and each of these inputs have a corresponding weight w1 through WN now what we can do is with each of these inputs and each of these ways we can multiply them correspondingly together and take a sum of all of them then we take this single number that's summation and we pass it through what's called a nonlinear activation function and that produces our final output Y now this is actually not entirely correct we also have what's called a bias term in this neuron which you can see here in green so the **bias term the purpose of the bias term is really to allow you to shift your activation function to the left and to the right regardless of your inputs right.**  Notice that the bias term doesn't is not affected by the X's it's just a bias associate to that input now on the right side you can see this diagram illustrated mathematically as a single equation and we can rewrite this as a linear using linear algebra in terms of vectors and dot products so instead of having a summation over all of the X's setting you'll see these kind of scattered in throughout the slides no need to really take furious notes at these code blocks.  The point of an activation function is to introduce nonlinearities into the data and this is  actually, really important in real life because in real life almost all of our data is nonlinear. |
| Sigmoid function- what is the sigmoid function? Definition  Probabilities connection to sigmoid function with values between 0 and 1. | Collapse X into a vector capital X which is now just a list or a vector of numbers a vector of inputs and also have a vector of weights capital W.  To compute the output of a single perceptron take the dot product of X and W which represents that element wise multiplication and summation and then apply that non-linearity which here is denoted as G.  What this nonlinear activation function is?  One common example of this activation function is what is called a  **sigmoid function.**  **A sigmoid function on the bottom right this function takes any real number as input on the x-axis and it transforms that real number into a scalar output between 0 & 1 it's a bounded output between 0 &** 1.  One very common use case of the sigmoid function is to when you're dealing with probabilities because probabilities have to also be bounded between 0 & 1 so sigmoid are really useful when you want to **output a single number and represent that number as a probability distribution in fact there are many common types of nonlinear activation functions** not just the sigmoid but many others that you can use in neural networks and here are some common ones and throughout this presentation |
| Summary |  |
| An introduction to Deep Learning and brief description of big data. Due to nonlinear data we can design and input everything into computer. Link between probability and sigmoid function. | |
| *Dorina Grossu*  *Professor*  *Conestoga College*  *dgrossu@conestogac.on.ca* | |