Project Report 1

Rahul Ramesh

Starting the project and Neural Networks

02-16-2019

1 Starting the project

This report will be broken down by what I've done to start the project, including setting up my coding environments and getting the data. Then the report will go into what I've learned about Neural networks and how they work including convolution neural network which is what I plan on using for the project. So lets being, I started the project 2 weeks ago, and progress has been good. First I set up my computational environment for this project.

1.1 Setting up my environments

The first thing that I did is set up a github for this project, so as to keep track of my code. The repository is available here: `https://github.com/rahram2010/Deepprotein`. Next I have set up my environments on my computer. I installed Anaconda, which is a environment manager on my computer to keep various projects serparate. After downloading and installing anaconda properly I made a environment called Deepprotein in my local file system, and this is the environment where I will do my work. I also installed all the necessary packages to that environment including Tensorflow, Keras, and jupyter notebook. These packages for python will allow me to write code more easily. Configuring these environments took a couple of days as I predicted to configure so that they work together, but they are working now and I can proceed to gather and clean data.

1.2 Gathering and Cleaning Data

Gathering Data proved more cumbersome than I previously anticipated. My ideal dataset would include a protein sequence, the appropriate GO functional labels and be large enough so that it can be usable. I first started searching the Uniprot website for an adequate dataset. The Swiss prot dataset seemed like a good match because it was both large enough(¿500,000 proteins) and contained the raw protein sequences. However getting the GO labels was not obvious. At first I thought I needed to write individual queries for each protein so as to get the GO labels, but this task for over 500,000 proteins would take too long. I then checked the GO website for more information.

The GO website is large and contains many different types of data including protein label data. This however was fruitless because the data file was not clear and using it to get non-redundant GO labels did not lead me anywhere. I looked for alternative datasets but none of the other datasets were large enough for my project, so I went back to the Swiss prot Dataset. I discovered from a bioinformatics discussion board website that you can filter the Swiss Prot dataset to include the GO dataset by doing an API call and retrieving the data that I wanted.

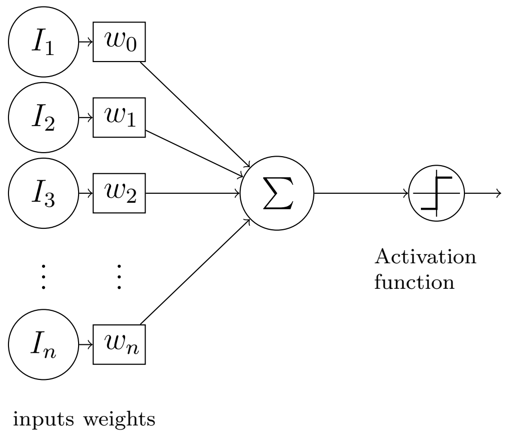
So my data consists of 10 columns–Entry, EntryName, Protein Names, Gene Names, Organism, Length, Sequence, GO, Status, Organism ID, Keywords. There is an example of the raw data that I will be using at the end of this report. After downloading the data, I have begun cleaning the data by removing redundant entries, and cleaning up by only using the ones that have valid experimental labels and removing ones that have empty function annotations which has left me with around 100 thousand data entries. Now I am ready to start coding the project, and building the model.

2 Neural Networks

I will start by explaining what I have learned about Nueral Networks from the basic level. This information is taken from the Bishop book and also online sources.

2.1 Single Layer Perceptron

A single layer perceptron is one of the most basic neural networks as it only has one layer, yet it illustrates the concept and some details fairly well. We define n inputs as $I_1, I_2, I_3, ..., I_n \in R^n$ and define n number of weights $w_1, w_2, w_3, ..., w_n \in R^n$. A single layer of a neural network works as such. Each input is multiplied by the weight and then they are passed to a processing unit, and after they are passed to a activation function or threshold function which "activates" or outputs 1 if the threshold is met. So this is the diagram of this network:



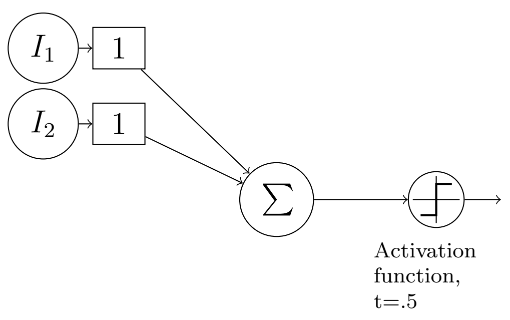
So a single layer can described as the weighted sum over the inputs, or $\sum_{i=0}^{n} w_i I_i$. And the most the most common activation function is the sigmoid function:

$$\frac{1}{1+e^{-x}}$$

It is important to note that the weights can be updated in an iterative way to minimize some loss function, and this is how the model gets better at the task. Typically for such a simple network the loss function would be the mean squared error. And how the algorithm would choose new weights would be through some optimization algorithm such as Gradient Descent.

2.2 Illustrative example

I will show the logical OR gate as a single layer neural network. The OR gate is such that if either of the inputs are 1 then the gate is 1. So we make this into a neural network.



inputs weights

Where our activation function is simply a threshold of $t = .5$. This neural network models a OR gate because the weighted sum of two inputs 0,0 is clearly 0 which is below .5 so the result is 0, and if either of the two inputs are 1 then they are above the threshold and the output is 1, and clearly when both of the inputs are 1 then the weighted sum is 2, and the output is 1.

2.3 Multiple Layer Perceptron

The multiple layer perceptron is put simply when single layer output is the input to another single layer perceptron and these two single layer perceptrons form a 2 layer perceptron and we can repeat this process for however many layers as our computational power allows. We can succinctly represent the single layer as follows

$$\sum_{i=0}^{n} w_i I_i$$

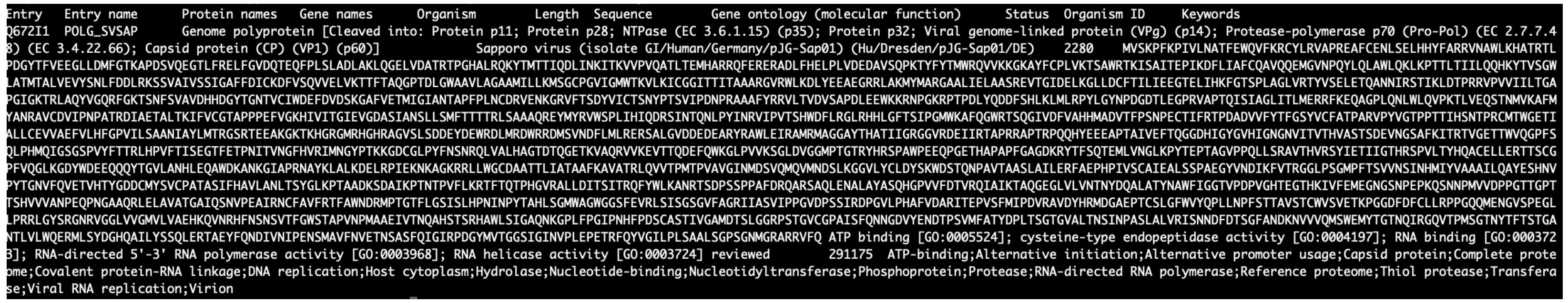
and we can represent the 2 layer perceptron as follows:

$$f(x) = \sigma(\sum_{j=0}^{m} w_j^2 (\sum_{i=0}^{n} w_i I_i))$$

where w^2 represents the second set of weights for the second layer, and the σ is the activation function. In the multiple layer neural network the output can be a probability distribution, and the activation function is nonlinear. There are many types of multilayer neural networks with many different types of activation functions. One problem that can occur with neural networks is that the presence of local minima in the error or loss function with the corresponding randomly assigned weights can throw off the invariance of the network. So one network that can overcome this problem without using regularization techniques is the Convolutional Neural Network.

2.3.1 Convolutional Neural Network

A convolutional Neural Network is exactly the same as the fully connected multilayer network described above except that each neuron is not connected to every neuron because this results in too complex network that depending on the activation function is not invariant, so the neuron is typically only connected to 3 other neurons in the next layer.



```
Entry   Entry name      Protein names   Gene names      Organism        Length  Sequence        Gene ontology (molecular function)      Status  Organism ID     Keywords
Q672I1  POLG_SVSAP      Genome polyprotein [Cleaved into: Protein p11; Protein p28; NTPase (EC 3.6.1.15) (p35); Protein p32; Viral genome-linked protein (VPg) (p14); Protease-polymerase p70 (Pro-Pol) (EC 2.7.7.4
8) (EC 3.4.22.66); Capsid protein (CP) (VP1) (p60)]             Sapporo virus (isolate GI/Human/Germany/pJG-Sap01) (Hu/Dresden/pJG-Sap01/DE)    2280    MVSKPFKPIVLNATFEWQVFKRCYLRVAPREAFCENLSELHHYFARRVNAWLKHATRTL
PDGYTFVEEGLLDMFGTKAPDSVQEGTLFRELFGVDQTEQFPLSLADLAKLQGELVDATRTPGHALRQKYTMTTIQDLINKITKVVPVQATLTEMHARRQFERERADLFHELPLVDEDAVSQPKTYFYTMWRQVVKKGKAYFCPLVKTSAWRTKISAITEPIKDFLIAFCQAVQQEMGVNPQYLQLAWLQKLKPTTLTIILQQHKYTVSGW
LATMTALVEVYSNLFDDLRKSSVAIVSSIGAFFDICKDFVSQVVELVKTTFTAQGPTDLGWAAVLAGAAMILLKMSGCPGVIGMWTKVLKICGGITTITAAARGVRWLKDLYEEAEGRRLAKMYMARGAALIELAASREVTGIDELKGLLDCFTILIEEGTELIHKFGTSPLAGLVRTYVSELETQANNIRSTIKLDTPRRVPVVIILTGA
PGIGKTRLAQYVGQRFGKTSNFSVAVDHHDGYTGNTVCIWDEFDVDSKGAFVETMIGIANTAPFPLNCDRVENKGRVFTSDYVICTSNYPTSVIPDNPRAAAFYRRVLTVDVSAPDLEEWKKRNPGKRPTPDLYQDDFSHLKLMLRPYLGYNPDGDTLEGPRVAPTQISIAGLITLMERRFKEQAGPLQNLWLQVPKTLVEQSTNMVKAFM
YANRAVCDVIPNPATRDIAETALTKIFVCGTAPPPEFVGKHIVITGIEVGDASIANSLLSMFTTTTRLSAAAQREYMYRVWSPLIHIQDRSINTQNLPYINRVIPVTSHWDFLRGLRHHLGFTSIPGMWKAFQGWRTSQGIVDFVAHHMADVTFPSNPECTIFRTPDADVVFYTFGSYVCFATPARVPYVGTPPTTIHSNTPRCMTWGETI
ALLCEVVAEFVLHFGPVILSAANIAYLMTRGSRTEEAKGKTKHGRGMRHGHRAGVSLSDDEYDEWRDLMRDWRRDMSVNDFLMLRERSALGVDDEDEARYRAWLEIRAMRMAGGAYTHATIIGRGGVRDEIIRTAPRRAPTRPQQHYEEEAPTAIVEFTQGGDHIGYGVHIGNGNVITVTHVASTSDEVNGSAFKITRTVGETTWVQGPFS
QLPHMQIGSGSPVYFTTRLHPVFTISEGTFETPNITVNGFHVRIMNGYPTKKGDCGLPYFNSNRQLVALHAGTDTQGETKVAQRVVKEVTTQDEFQWKGLPVVKSGLDVGGMPTGTRYHRSPAWPEEQPGETHAPAPFGAGDKRYTFSQTEMLVNGLKPYTEPTAGVPPQLLSRAVTHVRSYIETIIGTHRSPVLTYHQACELLERTTSCG
PFVQGLKGDYWDEEQQQYTGVLANHLEQAWDKANKGIAPRNAYKLALKDELRPIEKNKAGKRRLLWGCDAATTLIATAAFKAVATRLQVVTPMTPVAVGINMDSVQMQVMNDSLKGGVLYCLDYSKWDSTQNPAVTAASLAILERFAEPHPIVSCAIEALSSPAEGYVNDIKFVTRGGLPSGMPFTSVVNSINHMIYVAAAILQAYESHNV
PYTGNVFQVETVHTYGDDCMYSVCPATASIFHAVLANLTSYGLKPTAADKSDAIKPTNTPVFLKRTFTQTPHGVRALLDITSITRQFYWLKANRTSDPSSPPAFDRQARSAQLENALAYASQHGPVVFDTVRQIAIKTAQGEGLVLVNTNYDQALATYNAWFIGGTVPDPVGHTEGTHKIVFEMEGNGSNPEPKQSNNPMVVDPPGTTGPT
TSHVVVANPEQPNGAAQRLELAVATGAIQSNVPEAIRNCFAVFRTFAWNDRMPTGTFLGSISLHPNINPYTAHLSGMWAGWGGSFEVRLSISGSGVFAGRIIASVIPPGVDPSSIRDPGVLPHAFVDARITEPVSFMIPDVRAVDYHRMDGAEPTCSLGFWVYQPLLNPFSTTAVSTCWVSVETKPGGDFDFCLLRPPGQQMENGVSPEGL
LPRRLGYSRGNRVGGLVVGMVLVAEHKQVNRHFNSNSVTFGWSTAPVNPMAAEIVTNQAHSTSRHAWLSIGAQNKGPLFPGIPNHFPDSCASTIVGAMDTSLGGRPSTGVCGPAISFQNNGDVYENDTPSVMFATYDPLTSGTGVALTNSINPASLALVRISNNDFDTSGFANDKNVVVQMSWEMYTGTNQIRGQVTPMSGTNYTFTSTGA
NTLVLWQERMLSYDGHQAILYSSQLERTAEYFQNDIVNIPENSMAVFNVETNSASFQIGIRPDGYMVTGGSIGINVPLEPETRFQYVGILPLSAALSGPSGNMGRARRVFQ ATP binding [GO:0005524]; cysteine-type endopeptidase activity [GO:0004197]; RNA binding [GO:000372
3]; RNA-directed 5'-3' RNA polymerase activity [GO:0003968]; RNA helicase activity [GO:0003724] reviewed        291175  ATP-binding;Alternative initiation;Alternative promoter usage;Capsid protein;Complete prote
ome;Covalent protein-RNA linkage;DNA replication;Host cytoplasm;Hydrolase;Nucleotide-binding;Nucleotidyltransferase;Phosphoprotein;Protease;RNA-directed RNA polymerase;Reference proteome;Thiol protease;Transfera
se;Viral RNA replication;Virion
```