Using A Neural Network Simulator To Predict And Model The Spread Of A Pathogen

How can a neural network be created to predict the spread of a pathogen?

An IB Computer Science Extended Essay

3802 Words

Table of Contents

1. Introduction

Computer Science is an old field yet is consistently at the forefront of humanity's advancements; computers have become an important asset and tool from everyday life to important scientific projects. Computer simulations have seen a recent surge of advancements, with innovations and releases in graphical processing units and cloud services to complete ever-complex calculations and ease the burden on existing computers. Artificial Intelligence has also seen a resurgence in the field, with many computer scientists aiming to create more advanced AI technologies, although the aim of this paper focuses on a subset of the field known as neural networks.

Computer simulations and neural networks can be combined to result in more effective and possible more accurate predictions. Simulations work by having a set of formulas and calculations, taking inputs and outputting outputs. Neural networks can be taught through many methods, but generally the simplest explanation is that they are given a training dataset which are examples that the network can learn from. They are then given a test data set which the scientists can use to evaluate the network's predictive accuracy. By utilizing a neural network to simulate events, it could be more efficient and possibly have a higher accuracy in predictions.

How can a neural network be created to predict the spread of a pathogen? This paper serves to investigate if a neural network can be used to model a pathogen's spread, with the pathogen in this case being SARS-CoV-2, and attempt to create such a network. Given the pandemic situation regarding the novel coronavirus, an accurate neural network used to predict its spread could prove useful. Viruses are generally unpredictable, with surveillance and prevention of their spread being the only protection humanity has against a potential new pandemic. A neural

network that can be trained to predict the spread of pathogens with a high accuracy would help in creating new strategies to combat new molecular threats. Furthermore, new advancements in the field of artificial intelligence can also indirectly help many other industries such as the medical field or network security.

2. Background Information

As my only programming experience lies in web development with languages that do not work with neural networks, I had to construct a plan on how to complete such a complex task as a complete novice. A teacher recommended some sources and helped outline a general plan. The neural network would be built on Python and utilize Google's TensorFlow to assist with building the model, and use external libraries to visualize the results, so I would have to learn how to use those three. However, there were several choices to be made when arriving at this plan.

2.1 **Uses For Artificial Intelligence**

Artificial Intelligence as a field is extremely large, with subsets such as machine learning, deep learning, and neural networks. It is as widespread as it is known in the Computer Science field, seeing uses in GPUs to improve performance, enhancing images through AI upscaling, or even in fields such as agriculture to help farmers have better yields. With its widespread use and daily advancements, it could be possible to create a neural network that can sufficiently predict and model the spread of a pathogen. By building a network to model a brain, it allows for more flexibility and adaptability which leads to such a variety of uses.

2.2 **Why Use A Neural Network Over A Simulation**

Firstly, the choice to make a neural network as opposed to a simulation or algorithm was an important one. It was decided to choose neural networks because they can be molded. In a neural network, the inputs are known whereas the model's functioning itself does not need to be known. The simulation, however, requires you to understand everything so when it's made it will produce accurate results. Using real-world data for my training dataset for the neural network would also help it form into a more accurate prediction tool, as long as there was enough training data to use to teach it.

2.3 **Why Python Over R**

Python and R are both used in neural network computing. They both have benefits and disadvantages, such as R being notable for its use in statistical programming. Python was chosen as it: already bears similar syntax to Java and other languages I'm familiar with, has better integration with other languages and libraries that will allow me to visualize the data properly, has a large amount of libraries to use because of how widely the language itself is used, handles large amounts of data much better than R, and is a lot more flexible. (Ven) While R seems like a better language at first, as it is known for its statistical use, it seemed to be too difficult to spend time learning and using when Python was a simpler and easier to use alternative.

2.4 **Other Tools**

There are other machine learning libraries available to use and help train the neural network. The two most popular ones that appeared when researching are Keras and TensorFlow. I chose TensorFlow at first because it is open source and is more flexible in terms of high and low-level

APIs. (Choudhury) There are also more tutorials and resources that I could find, as well as my teacher being familiar with it already. However, as I began the project, I realized that I might as well use both. There is nothing preventing you from utilizing both libraries and they had resources that proved to be beneficial.

2.5 **Visualizing the Dataset**

From the start, the problem of visualizing the results was there. It is important to make the predictions easy to understand and detailed enough to be worth making a neural network. The original idea was to use MATLAB as the school offered it, and there were resources that were provided to teach me. However, the school closed, and I ended up attempting to use libraries such as NumPy, pandas, and Matplotlib to work with the data and visualize it. This was also scrapped in the end as it never turned out to be successful, but the results will be covered in the results section.

2.6 **Why the Novel Coronavirus**

The choice to use the novel coronavirus was easy, as it is a current problem but also a highly recorded event. Using data gathered from its spread in my training dataset, I can model the neural network to be more accurate as there is so much data to feed it. There are multiple sources that I can get my information from, though I will be mainly relying on a dataset provided by Our World in Data. It is updated daily and has total case and death numbers in each country that has reported them. It also has the benefits of providing the data in a .csv, or comma separated values, file format which is easier to parse through and use with Python.

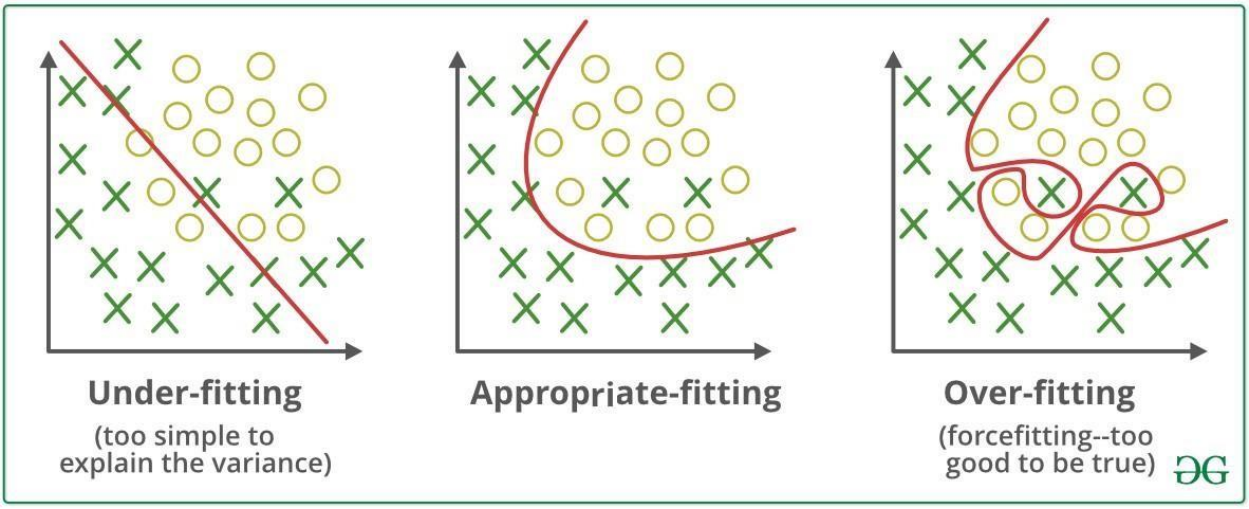
2.7 **Potential Challenges and Difficulties**

Although the implementation of a neural network which utilizes a multitude of datasets appears to have the potential to provide an objective and unbiased approximation of observed trends, the concept of bias is nevertheless present in any machine learning training dataset. Aside from the numerous complexities and difficulties involved with the initial setup such as learning the languages and how to utilize them and their tools properly (artificial intelligence and neural networking being almost a separate field of study), the collection and distribution of diverse datasets, and the optimization and creation of an efficient neural network to operate without bias proved to be challenging tasks.

2.8 **Mitigations of Tribulations**

"That the more simple any thing is, the less liable it is to be disordered, and the easier repaired when disordered" – Thomas Paine

As inherent with the problem of artificial intelligence, the optimization of any algorithm involves the control of its complexities. Thus, trial and error processes through experimentation may perhaps yield the only method to simplify an algorithm or its inner mechanisms. One notably observable complexity in the aforementioned section was bias. In its simplest terms, bias may be recognized as an element which influences and skews results. Based on the premise of this definition, the particular application of bias in the scope of machine learning may be regarded as the lack of precision or accuracy in an algorithm's resulting interpretation of data.



For instance, consider the figure above – the instance that is demonstrated on the far right represents bias through ‘over-fitting’. It forces itself to follow the trend too closely, which leads to an unrealistic output. In a sense, the algorithm conforms too strictly to the data provided which does not leave room to identify a general trend that may be present within a greater variety of related data. This commonly occurs when the machine learning algorithm trains too frequently on the same dataset. On the other hand, the instance demonstrated on the far left represents the concept known as ‘under-fitting’ which is an issue prevalent when an algorithm is not trained enough. Between these two, however, is the ‘appropriate-fitting’ instance which identifies a general trend among various data points – this is the optimal model. Hence, a trade-off is observable; it appears that the optimal model may only be achieved through a balancing training frequency with variety. In other words, the higher the threshold of accuracy by which one wishes a model to operate under, the greater the scale by which data must be sourced and the algorithm trained. In this specific project, appropriate fitting will be achieved through proportionally scaling the utilization of both datasets and training volume which would reduce both the inaccuracy and the overfitting of the data. Collected data will not be limited to merely a single source; for instance, the Johns Hopkins data, in conjunction with the excel spreadsheet and a

variety of other sources such as people from the CDC mentioned previously would provide a suitable number of data points to train the algorithm on with a considerable frequency without risking bias through over-fitting. Moving on to the subject of optimization, it is an essential practice to minimize the amount of superfluous and unnecessary calculations handled by the algorithm; this will maximize the scalability of the system when it comes to performing large calculations over any size of data. Although Python is already a highly efficient and optimized platform for machine (in comparison to other languages such as R), appropriately simplifying the modeling process will ensure that the program operates to its best ability. One common method by which the optimization of a model can be performed is through the reduction of non-correlating parameters. (Chan) In this scenario, if a variable such as gender does not affect the range of which the virus can spread, it is thus not necessary for the neural network to consider such factors. Because we do not necessarily know the factors which do not contribute to the modeling of the spread of the virus, these variables can be identified and isolated only through the means of trial and error. For instance, as seen in Matthew Chan's demonstration of balancing a cart-pole, a significant reduction of complexity in the modeling system was made through the removal of two variables. This optimization provided for the reduction of total steps from 681 to 136. As such, I can apply similar tactics in order to reduce the computing load of the neural network on my computer while also drastically reducing computing time. Although the removal of variables posits a risk in lack of accuracy, the aim of a neural network is to propose a render that is close to or significantly close to a real-life situation.

2.9 **How It Works**

Conceptually, each ‘node’ in a neural network acts like a neuron. They each contain a mathematical function and calculates a weighted sum from its inputs, with larger weighted inputs leading to larger weighted outputs. (Lee) This sum is then entered into a non-linear function called the activation function which allows the network to model complex and non-linear data. (Lee) The network can also be visualized as an input layer, a series of hidden layers, and an output layer. The hidden layers perform the calculations with the activation functions which will allow for approximated results which lead to the final predictions.

3. **The Making of The Neural Network**

3.1 **The Training Data**

Good training data is imperative for the effectiveness of the model. Since the goal is to predict the spread of the 2019-20 coronavirus, data must be accurate and extensive. The plan is that the inputs will include countries, case numbers, and days after first infections. Factoring in this data, the model should be able to be trained into inferring future case numbers given a country and length of time. Luckily, many sources exist online where such datasets exist. As stated earlier, I was able to use the Our World in Data dataset as it is updated daily and proved to be extremely extensive in its values. The website claims it only receives information from trusted sources such as the European Centers for Disease Control and Prevention and other official government releases. To ensure consistency, I cross checked it with other numbers such as from Google. I was able to edit it in Google Sheets and it had information from December 31 of 2019. This file was then reduced into columns of countries, days after the first case, and cases to start with as a proof of concept. Python has native support to read and write CSV files which proved to be useful in

order to train the model with the file. There are, of course, concerns with the data to cover. The training data itself may have biases that I would not be able to eliminate, such as a misrepresentation of numbers when a country reports them for any variety of reasons. Also, when considering the values, I do not think that there were any special values that had specific weights to make certain variables have more of an influence in the model as there were not many variables in the first place. This is an extremely simplified model that can still be improved on, and I am sure that length of time and designated country have the same weights.

3.2 **Implementation of Solution**

The model's construction should be relatively simple, given the fact that Google's TensorFlow libraries are extensive enough that the majority of the work can be imported rather than building a new model from scratch. In order to write the programs, Google Colab was used to make it easier to connect the code with everything. One of the original ideas, for example, was to connect the program with a geolocation API and retrieve latitude and longitude coordinates by giving a name of a place. This idea could've be helpful, but the API has the issue of being very slow when dealing with many places at once and had to be abandoned in favor of using the dataset.

3.2.1 **Libraries**

There are many Python libraries that exist to make dealing with machine learning models easier. In my case, I found a model that could be turned into something to fit my needs based off of Keras and TensorFlow which was even better. Other libraries that helped include NumPy, for computing, and Pandas for manipulating data. Instead of having to worry about weight and individual neurons in the hidden layers, it was as simple as using methods and choosing an activation function. The advantages of Python are emphasized here, with the number of libraries

available to lighten the workload and help with the work. Of course, the model and libraries I found still had to be edited to work with my goals for the project. The real issues began when trying to train the model, as deciding on what to change to fix the model was the real issue. Building the model was not the issue, but rather trying to understand what needed to be fixed in order to make it more accurate.

3.2.2 **Normalization of Data**

A major problem that was first encountered was the extremely skewed outputs given certain inputs. After some further research, I realized that the data points were too large. With case numbers reaching numbers in the millions, the weights that the AI would assign to different values would have less of an effect with regard to any consideration of the other parameters. This led to inconsistent outputs, which could be fixed with multiple methods, but one prominent solution was to normalize the datasets. Normalizing the data would make them less than 1 and greater than 0, allowing for smaller values that the model could work with and possibly produce more accurate predictions. I was able to find a normalization function and apply that to the inputs and outputs of the model to better calculate values. This was able to help somewhat, such as when calculating case numbers for Moscow, Russia where the predicted numbers were 10,000 off from being accurate.

3.2.3 **Model Used**

The model used in the project was based off one used to predict the equation $y=5x$. Since my model itself is a regression model, it was plausible to use the same libraries and methods. The only difference being that the spread of a virus is exponential instead of linear and the inputs are a

lot more different and varied as compared to a linear function. On an abstract level, the model implements a “sequential” neural network architecture which relies on the feeding of inputs which will be transformed through each sequential layer in the network – each of which performs a multitude of mathematical operations through units called neurons. Based on analysis of the three-dimensional data set (dates, location, and cases), it was concluded that multiple layers were needed in order to model the complex data. Within these hidden layers, the number of hidden neurons varied and had to be tweaked. There were not any obvious trends based on human analysis, so it was important to ensure that the model has the capacity to identify and model complex trends. Furthermore, the model had to balance the line, as discussed earlier, of not being overfitted while also not being underfitted. Epochs, or an instance of training over every data point, could be varied from 100 times the model was trained to 5000 times. Each variable had to be considered, although there were issues when attempting to isolate each segment of the model. There are methods to check for overfitting as well, such as checking the accuracy of the model with the testing set vs the training set. The model was accurate enough for the training and the testing sets. If it was overfit, it would have been extremely accurate in regard to the training set but inaccurate with the testing set.

4. Results

4.1 **Data Analysis**

Loss values in the program demonstrate the average deviation in the predicted value compared to the given value in the model. As the loss value increases, the error margin will also increase, which is not preferable. A loss value as close to 0 as possible is best. However, it is important to note that as the data is normalized and becomes smaller, so too does the loss value in proportion

with the data. It is important to keep this in mind, as the value would go from 32 with nonnormalized data points to 0.03 with normalized data points. With more training I found that the loss value would decrease to numbers as low as 0.0000003, which was much more preferable, even in proportion with the normalized data. In certain scenarios, the model after being trained would get as accurate as 10,000 cases off. While the number could certainly be more accurate, 10,000 cases was an acceptable margin for this basic AI. Of course, the model could still be fooled with false statistics and disproportionately high numbers which proved to be a weakness.

4.2 **Implications of the Project**

This project, with its ability to predict trends and model a virus's rate of infection and spread, has numerous important implications, including saving many lives. Through the model's ability to identify trends from old data, it can then provide assistance to scientists and researchers in terms of preventing the further spread of a future virus or pathogen and even setting deadlines for the creation of a new vaccine or treatment. Using this model to predict future infections and infection rates allows scientists to identify trends that perhaps a human could not achieve through normal methods. For example, if a future virus is similar to an older or current one, its spread can be predicted and effectively stopped – therefore making this process more effective than any traditional methods. The AI may also be able to identify trends found in older diseases, allowing scientists to learn paths and patterns that may not have been found previously. In short, the implications that such a model would have for the future of virology and medicine could be unprecedented if done right. Obviously, my model is extremely basic and needs a lot more refining; however, if truly done right, this concept has the potential to save the lives of many as humanity deals with increasingly dangerous microbial threats.

4.3 **Limitations faced**

Some of the most glaring issues became prominent when attempting to output the predictions for cases. Most problems faced were discovered in the building of the model, as changing the number of hidden neurons and layers could change outputs drastically. It became difficult to isolate single issues as there were many variables in the AI. Aside from inconsistencies found in high loss, training, or building, the most notable issue was found when comparing the model with certain countries and their reported numbers. For example, the predicted number for China was greatly larger than the reported number. This could have been due to the likely chance that China misreported its data, but it could also have been an error when training the machine. Of course, if such an AI were to be truly effective, the dataset should not be biased, and it would be largely more helpful if the case numbers were accurate. Lastly, there were the aforementioned issues with visualizing the data that can be improved on in the future.

4.4 **Conclusion**

So, **how can a neural network be created to predict the spread of a pathogen?** The model was accurate, to an extent, in some areas but led to a wide variability in others; again, most likely caused by inaccuracies and inconsistencies, whether intentional or nonintentional, in the numbers reported by countries. This was one error in the dataset that proved to be highly ineffectual in achieving the desired effect, but the model itself could be improved on in terms of length of training, repetitions of training, or even changing the model and its structure. A neural network would be overall successful in achieving the desired effect of modeling the spread of the novel coronavirus, but I would recommend performing an exponential regression on the data instead, rather than attempting to build a neural network. The math could have been done with regular

mathematics instead of making a machine learning model. This foray into machine learning was beneficial and opened up many new doors, and future development on machine learning in general will prove to be an advantage for researchers and scientists of any field everywhere.

Works Cited

Authors. “Hidden Layer.” *DeepAI*, DeepAI, 17 May 2019, deepai.org/machine-learningglossaryand-terms/hidden-layer-machine-learning.

binga. “Python vs R for Machine Learning.” *Python vs R for Machine Learning*, Stack Exchange, 17 Jan. 2017, datascience.stackexchange.com/questions/326/python-vs-rformachine-learning.

Chan, Matthew. “Cart-Pole Balancing with Q-Learning.” *Medium*, Medium, 13 Nov. 2016, medium.com/@tuzzer/cart-pole-balancing-with-q-learning-b54c6068d947.

Choudhury, Ambika. “TensorFlow vs Keras: Which One Should You Choose.” *Analytics India Magazine*, Analytics India Magazine, 28 June 2019, analyticsindiamag.com/tensorflowvskeras-which-one-should-you-choose/.

“Getting Started with Keras (AI Adventures).” *YouTube*, YouTube, 14 Aug. 2018, www.youtube.com/watch?v=J6Ok8p463C4.

Lee, Timothy B. “How Neural Networks Work-and Why They've Become a Big Business.” *Ars Technica*, Ars Technica, 2 Dec. 2019, 8:00, arstechnica.com/science/2019/12/howneuralnetworks-work-and-why-theyve-become-a-big-business/.

Nain, Aakash. “TensorFlow or Keras? Which One Should I Learn?” *Medium*, Imploding Gradients, 13 May 2017, medium.com/implodinggradients/tensorflow-or-keras-whichoneshould-i-learn-5dd7fa3f9ca0.

Paine, Thomas. *Common Sense*. January 10 1776

Ritchie, Hannah. “Coronavirus Source Data.” *Our World in Data*, Our World in Data, ourworldindata.org/coronavirus-source-data.

“Simulating an Epidemic.” *YouTube*, YouTube, 27 Mar. 2020, www.youtube.com/watch?v=gxAaO2rsdIs.

“Tutorials : TensorFlow Core.” *TensorFlow*, Google, www.tensorflow.org/tutorials.

Ven, Raj. “Python vs. R: Which Should You Choose For Your Next ML Project? - DZone AI.” *Dzone.com*, DZone, 13 Aug. 2018, dzone.com/articles/python-or-r-which-shouldyouchoose-for-your-next.