Crime Prediction using Deep Learning

Final Report
ELE494-09

Nasir Khalid
b00065082

Yousif Khaireddin
b00063618

***Abstract*—The abstract goes here.**

I. Introduction

Around the world police departments from different regions invest large amounts of money in finding ways to discover crime trends, uncover potential crime plans and develop better policing techniques. In the early days of crime prediction and technology, this was mainly done by observing historical data to find common trends in crime over years, months or even days. Apart from this a lot of undercover officers were used to patrol and be on the lookout for suspicious ongoings in the city. These methods were often very expensive and ignored certain factors that affect crime.

Crime itself is unpredictable in nature when viewed on its own and it is shown to be dependent on various different factors such as weather, location, social and economic factors [1] When prediction was done using historical data these factors were ignored and therefore the models did not perform well. With the advancement of deep learning and high powered computers a new method of crime prediction became more viable, a method that was capable of finding links between various other factors and crime. However, crime prediction is a field that has not received the same level of attention from deep learning as other fields like computer vision, generative modeling etc.

Deep neural networks are designed to reduce the need for extensive feature engineering and allow for training over large datasets. The deep entanglement of crime, multiple variables and it's dependency on spatial and temporal factors (location and time of day) make it an ideal candidate for prediction with a deep neural network.

II. Literature Review

One of the earliest cases of crime prediction through a neural network used regression to predict the monthly 911 calls and thereby predict crime [2]. However, this network was limited by technology and was not capable of using location and other factors for prediction.

Another recent method was to divide the city in to a grid like a checkerboard and data is aggregated within each cell. These cells also contain previous days crime as well as information about the neighboring cell, this was used to then predict the average number of crimes over a month and the type of crimes within a month [3]

The most recent paper on this topic [4] once again used a grid like division of the city like the one in [3] however they instead used a RNN and CNN to train the network to account for temporal and spatial information instead of adding crime data from previous days and using the neighboring cell data.

III. Method

For our project we wanted to step away from the methods used in [4] and [3] that relied on splitting a city in to cells and training networks only on these cells. Instead we wanted to utilize the existing location details available in many datasets such as co-ordinates and neighbourhoods to predict crime. To do this we created multiple different networks that all utilized data that was available in our datasets and did not need divisions of a city in to grid blocks.

A. Datasets Used

For the project we obtained all of our data from the city of Vancouver's open data source [5]. Listed below are all the datasets we used in our project, their specific use cases are discussed in detail in further sections.

1) Crime Data:

The first dataset we obtained from the website was the crime dataset [6]. The columns of the dataset were the following:

- Type of crime
- Year
- Day
- Month
- Hour
- Minute
- Block of crime
- Neighbourhood of crime
- X Co-ordinate of crime in UTM Zone 10
- Y Co-ordinate of crime in UTM Zone 10
- Latitude
- Longitude

The entire dataset consisted of 530652 crimes from 2003 to 2017. However, for some of the them their time and location data was missing as it was protected for privacy reasons. This data was essentially useless for us so we eliminated all of these crimes and that left us with a total of 476290 crimes to work with.

Shown below is a figure showing the number of crimes per day from 2004 to 2017 obtained from the dataset:

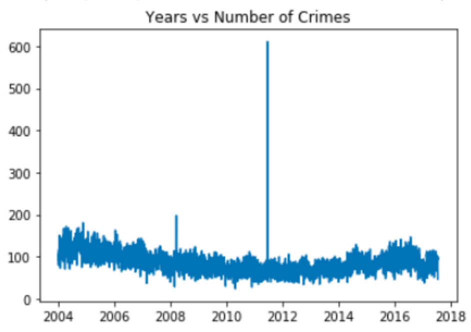


Figure 1: Crimes per day from 2004 to 2017

2) Neighbourhood Dataset:
We then downloaded a second dataset from the Vancouver open data catalogue that gave us a list of neighborhoods in Vancouver [7]. We added 2 new columns to this dataset called 'Latitude' and 'Longitude' and in here we added the center latitude and longitude for each respective neighborhood. This second dataset consisted of the following columns:

- Map ID
- Neighbourhood Name
- Neighbourhood Center Latitude (Added)
- Neighbourhood Center Longitude (Added)

3) Graffiti Dataset:
A list of 8508 points detailing the exact location of all known graffiti in Vancouver [8]. The dataset has the following columns:

- Latitude
- Longitude

4) Water Fountain Dataset:
A list of 241 drinking fountains scattered around Vancouver [9] containing specific information about each of them, the dataset has the following columns:

- Map ID
- Latitude
- Longitude
- Name
- Location
- Maintainer
- In Operation
- Pet Friendly
- Photo

For our project we only rely on the latitude and longitude information from the dataset, the other columns are discarded.

5) Google Search Trends of Crime:
A monthly index describing the rate at which the word "crime" has been searched throughout a month relative to that of the maximum month [10]. This dataset spans from 2004 till present, so it requires the removal of 1 year of the crime data (2003) during merging.

We use this dataset because during our research we learned that there is a very high correlation between the 6-Months Moving Average moving average of google searches for crime and the 6-Months Moving Average of crimes in Vancouver, this can be seen in the figure below:

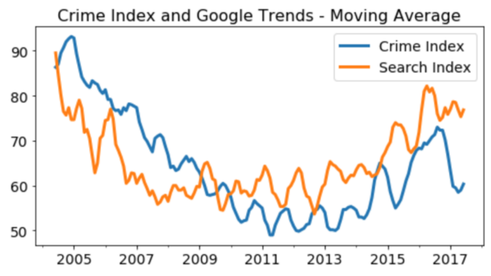


Figure 2: 6 Month moving average of Crime and Crime Search

We decided to include this metric of google searches into our final dataset. We decided that for each datapoint, we would insert the google crime search trend of the previous month, and the code for including this to the final dataset can be found on the project repository. [11] One important thing to notice is that this is a form of temporal dependency since it is indirectly describing the behaviors of this dataset for each entry over time.

B. Final Dataset

It is important to note that all the previously listed datasets waere then further filtered and manipulated to fit the needs of each and every network we built, and this will be discussed in the respective network sections. Some networks utilized more datasets than others, and this was dependent on the nature of the problem at hand, as will be described later. However, we did compile all available data and create a final crime dataset [12] that was imported to each networks respective Jupyter notebook. It consists of the following columns:

- Type of Crime
- Year

- Month
- Day
- Hour
- Minute
- Latitude
- Longitude
- Neighbourhood
- Block of Crime
- X Co-ordinate of crime in UTM Zone 10
- Y Co-ordinate of crime in UTM Zone 10
- Distance from nearest Graffiti
- Distance from nearest Drinking Fountain
- Google Trend Data for previous month

1) Distance from nearest Graffiti and Drinking Fountain: These columns in the final dataset were added manually by us. We first looped over all crimes and for each crime we calculated its distance in kilometres from each of the different graffiti locations and each of the different drinking fountain locations. We then got the smallest value for each calculated distance and added it to the dataset. This was done to account for spatial features that consider the presence of nearby factors that could lead to crime. The script used for this process can be found on the project repository [13] [14]

Shown below is a histogram of the distances from crimes and it is clear that crimes happen frequently near graffiti and around a certain distance away from drinking fountains.

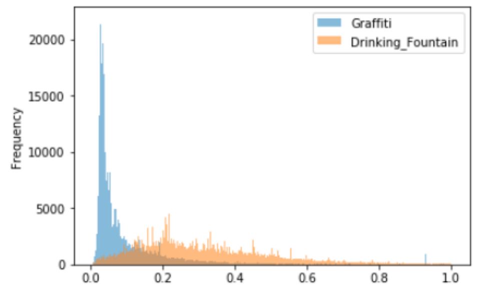


Figure 3: Histogram of Distance from Graffiti and Distance from Drinking Fountain

C. Training Plan

Now that we had collected our dataset with multiple different variables our plan was to create multiple different networks with different variations and see the results that we obtain from each of these networks. We trained various deep neural networks and these are outlined in the upcoming sections.

IV. Experiments

A. Network 1 [15]

For one of our very first networks, we decided to create a sum of total crimes per neighborhood per day. This can be done by first up-sampling the crime dataset to include all neighborhoods in each day even if there was no crime in the neighborhood, then we sum the number of crimes per day per neighborhood. So per day there are 22 entries (one for each neighborhood) and there is a new column with the total number of crimes. A graph of this can be seen below:

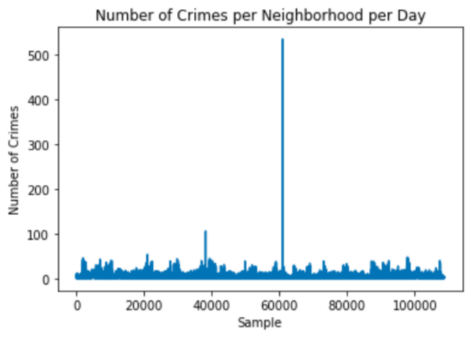


Figure 4: Number of Crimes per Neighbourhood per day

Once the above has been completed, we can then pool these different crimes into bins which denote a range of crimes. This is important as it will enable us to easily one hot encode the crime values into a number of classes. This is useful as it reduces the complexity of the problem. For our application, we chose the following ranges:

- Class 0: 0 – 1 crimes
- Class 1: 2 - 3 crimes
- Class 2: 4 - 5 crimes
- Class 3: Greater than 5 crimes

A graph displaying this information can be seen below:

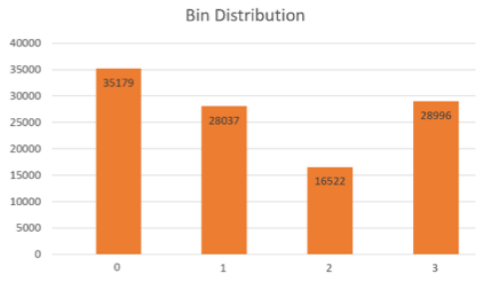


Figure 5: Bin Distribution

All of this was compiled in to a dataset that is available on the project repository [16] and the code used to perform this is also available [17]

Once this data has been formulated, we can then run this information through a neural network. For this task, we decided to build a simple feedforward network since there is no clear method to map this information into a CNN or RNN.

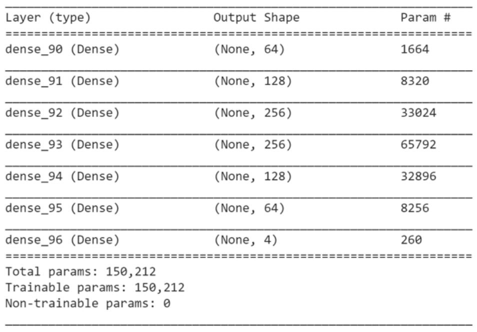
The network inputs:

- Year
- Month
- Day
- Neighbourhood

The network output:

- A vector with a probability associated to each bin

We used the following network architecture which had the best performance:



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_90 (Dense)             (None, 64)                1664
_________________________________________________________________
dense_91 (Dense)             (None, 128)               8320
_________________________________________________________________
dense_92 (Dense)             (None, 256)               33024
_________________________________________________________________
dense_93 (Dense)             (None, 256)               65792
_________________________________________________________________
dense_94 (Dense)             (None, 128)               32896
_________________________________________________________________
dense_95 (Dense)             (None, 64)                8256
_________________________________________________________________
dense_96 (Dense)             (None, 4)                 260
=================================================================
Total params: 150,212
Trainable params: 150,212
Non-trainable params: 0
_________________________________________________________________
```

Figure 6: Network 1 Architecture

The following two plots show the accuracy and loss of our network throughout the training:

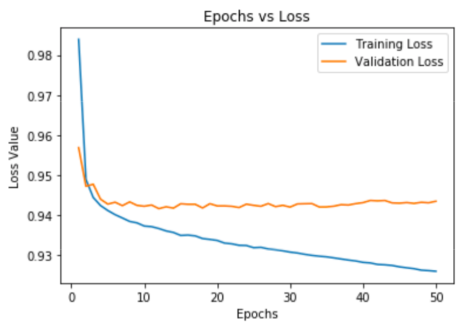


Figure 7: Network 1: Epoch vs Loss

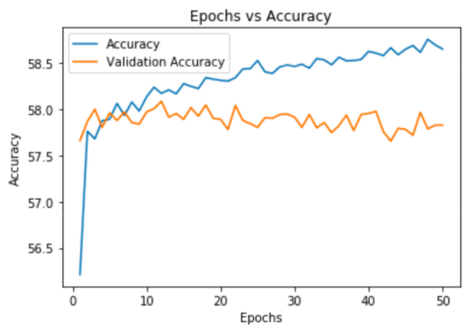
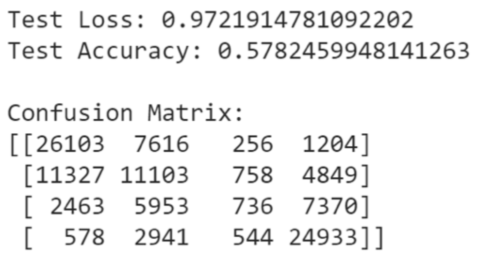


Figure 8: Network 1: Epoch vs Accuracy

When used on the test data, the network had a final test loss of 0.97219 and an accuracy of 58%. The confusion matrix is shown below:



```
Test Loss: 0.9721914781092202
Test Accuracy: 0.5782459948141263

Confusion Matrix:
[[26103  7616   256  1204]
 [11327 11103   758  4849]
 [ 2463  5953   736  7370]
 [  578  2941   544 24933]]
```

Figure 9: Network 1: Confusion Matrix

The network above is not ideal by any means; however, it is important to note that increasing the complexity of the network did not yield any better results. Next, it is important to note that since we are not overfitting the data, we did not put any means of regularization or dropout, as was seen in the network architecture above. This, however, does show that it might be possible to properly predict the number of crimes per neighborhood per day should we have a larger dataset, or some more features on this data.

B. Network 2 [18]

In this network we attempted to predict the type of crime that was most likely at a certain hour of day in a certain location. For this network we used the previously mentioned final dataset that we had prepared [12] as it contains all the needed data.

The network inputs:

- Year
- Month
- Day
- Latitude
- Longitude
- Distance to Graffiti
- Distance to a Drinking Fountain

The network output:

- A vector with a probability associated to each Type of Crime

The required columns were imported and the output was a one hot encoded vector related to each crime. Before training we also tried to understand which crimes happen most frequently as we understand that our network would return a higher probability related to those ones. Shown below is a histogram of the dataset showing the count for different types of crimes:

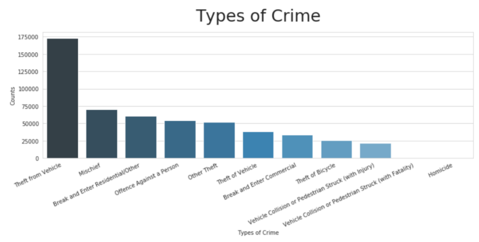
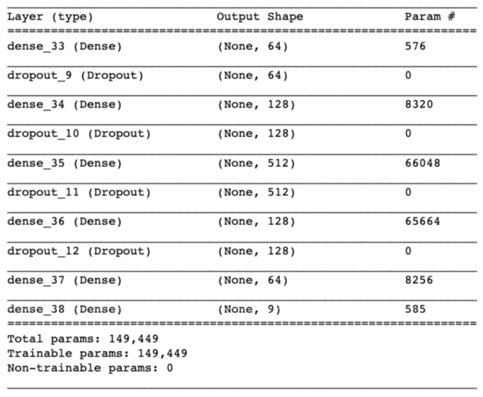


Figure 10: Histogram of types of Crime in city

We used the following network architecture which had the best performance:



```
Layer (type)                 Output Shape              Param #
=================================================================
dense_33 (Dense)             (None, 64)                576
_________________________________________________________________
dropout_9 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_34 (Dense)             (None, 128)               8320
_________________________________________________________________
dropout_10 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_35 (Dense)             (None, 512)               66048
_________________________________________________________________
dropout_11 (Dropout)         (None, 512)               0
_________________________________________________________________
dense_36 (Dense)             (None, 128)               65664
_________________________________________________________________
dropout_12 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_37 (Dense)             (None, 64)                8256
_________________________________________________________________
dense_38 (Dense)             (None, 9)                 585
=================================================================
Total params: 149,449
Trainable params: 149,449
Non-trainable params: 0
_________________________________________________________________
```

Figure 11: Network 2 Architecture

The following two plots show the accuracy and loss of our network throughout the training:

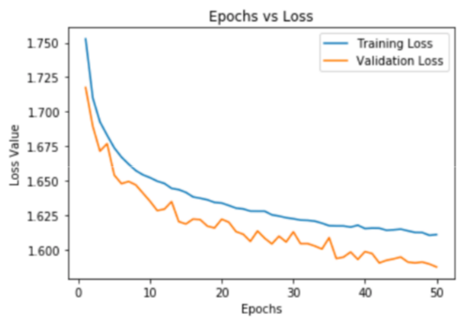


Figure 12: Network 2: Epoch vs Loss

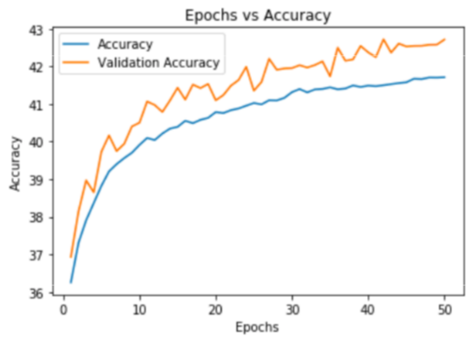
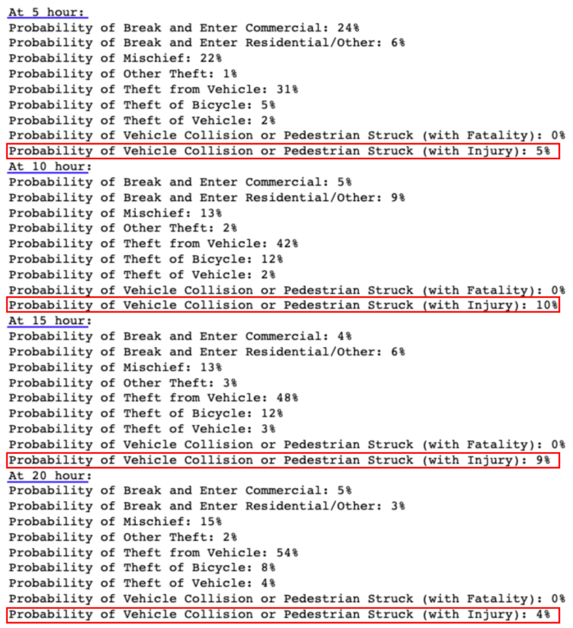


Figure 13: Network 2: Epoch vs Accuracy

On the test set the network had a 43% accuracy. Although our network did not perform well in terms of accuracy we still found that it had some sort of understanding in to the occurance of crime and times of day at a certain location. For example as we see the output of the network over 24 hours we see that it shows a lower chance of crime due to vehicle/pedestrain accident at night but more during rush hour. This does make sense and it shows that there is some understanding that our network is able to develop during training phase. There may be more cases like which are not explicitly visible at first glance. Shown below is an image displaying the results over 24 hours and the probability changing over time:



```
At 5 hour:
Probability of Break and Enter Commercial: 24%
Probability of Break and Enter Residential/Other: 6%
Probability of Mischief: 22%
Probability of Other Theft: 1%
Probability of Theft from Vehicle: 31%
Probability of Theft of Bicycle: 5%
Probability of Theft of Vehicle: 2%
Probability of Vehicle Collision or Pedestrian Struck (with Fatality): 0%
Probability of Vehicle Collision or Pedestrian Struck (with Injury): 5%
At 10 hour:
Probability of Break and Enter Commercial: 5%
Probability of Break and Enter Residential/Other: 9%
Probability of Mischief: 13%
Probability of Other Theft: 2%
Probability of Theft from Vehicle: 42%
Probability of Theft of Bicycle: 12%
Probability of Theft of Vehicle: 2%
Probability of Vehicle Collision or Pedestrian Struck (with Fatality): 0%
Probability of Vehicle Collision or Pedestrian Struck (with Injury): 10%
At 15 hour:
Probability of Break and Enter Commercial: 4%
Probability of Break and Enter Residential/Other: 6%
Probability of Mischief: 13%
Probability of Other Theft: 3%
Probability of Theft from Vehicle: 48%
Probability of Theft of Bicycle: 12%
Probability of Theft of Vehicle: 3%
Probability of Vehicle Collision or Pedestrian Struck (with Fatality): 0%
Probability of Vehicle Collision or Pedestrian Struck (with Injury): 9%
At 20 hour:
Probability of Break and Enter Commercial: 5%
Probability of Break and Enter Residential/Other: 3%
Probability of Mischief: 15%
Probability of Other Theft: 2%
Probability of Theft from Vehicle: 54%
Probability of Theft of Bicycle: 8%
Probability of Theft of Vehicle: 4%
Probability of Vehicle Collision or Pedestrian Struck (with Fatality): 0%
Probability of Vehicle Collision or Pedestrian Struck (with Injury): 4%
```

Figure 14: Network 2: Result over 24 hours with pedestrian/vehicle collision highlighted

C. Network 3 [19]

In this network we attempted to predict the probability of crime in a neighborhood at a certain hour of day. For this network we used the previously mentioned final dataset that we had prepared [12] as it contains all the needed data.

The network inputs:

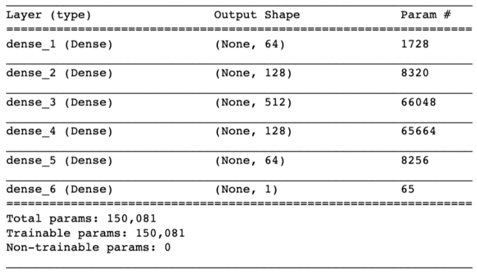
- Year
- Month
- Day
- Hour
- Neighbourhood (One hot encoded)

The network output:

- Probability of crime occuring

Here before training the data was first given a 'Crime' column with a value of 1 indicating that a crime happened. Then we performed upsampling. The data is upsampled by neighborhood so that if for some day/time there was no crime in a neighborhood, a point was added for that neighborhood and the 'Crime' column value was set to 0. We also upsampled by 1 hour so that if there was no crime at for example 6:00 am in a neighborhood, a row was added for that time and neighborhood with it's 'Crime' value set to 0.

We used the following network architecture which had the best performance:



```
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 64)                1728
_________________________________________________________________
dense_2 (Dense)              (None, 128)               8320
_________________________________________________________________
dense_3 (Dense)              (None, 512)               66048
_________________________________________________________________
dense_4 (Dense)              (None, 128)               65664
_________________________________________________________________
dense_5 (Dense)              (None, 64)                8256
_________________________________________________________________
dense_6 (Dense)              (None, 1)                 65
=================================================================
Total params: 150,081
Trainable params: 150,081
Non-trainable params: 0
_________________________________________________________________
```

Figure 15: Network 3 Architecture

The following two plots show the accuracy and loss of our network throughout the training:

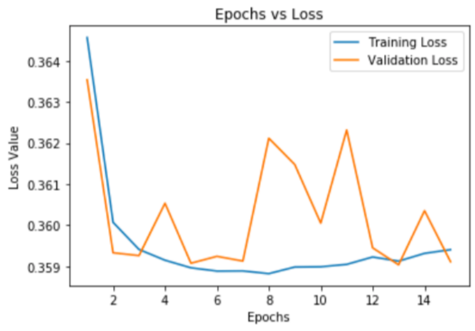


Figure 16: Network 3: Epoch vs Loss

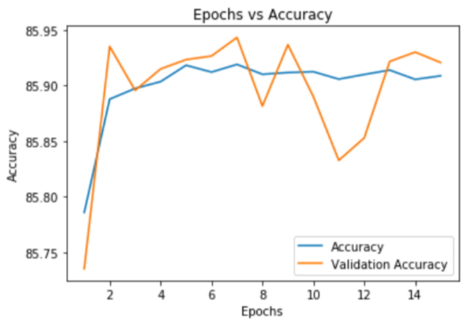
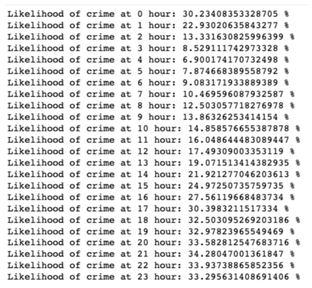


Figure 17: Network 3: Epoch vs Accuracy

On the test dataset the network had a final loss of 0.36 and an accuracy of 86%. The output for a given day and location over 24 hours is shown below and it does exhibit a logical pattern with crime being more likely to happen at night but not very likely around the early hours:



```
Likelihood of crime at 0 hour: 30.23408353328705 %
Likelihood of crime at 1 hour: 22.93020635843277 %
Likelihood of crime at 2 hour: 13.331630825996399 %
Likelihood of crime at 3 hour: 8.529111742973328 %
Likelihood of crime at 4 hour: 6.900174170732498 %
Likelihood of crime at 5 hour: 7.874668389558792 %
Likelihood of crime at 6 hour: 9.083171933889389 %
Likelihood of crime at 7 hour: 10.469596087932587 %
Likelihood of crime at 8 hour: 12.503057718276978 %
Likelihood of crime at 9 hour: 13.86326253414154 %
Likelihood of crime at 10 hour: 14.858576655387878 %
Likelihood of crime at 11 hour: 16.048644483089447 %
Likelihood of crime at 12 hour: 17.49309003353119 %
Likelihood of crime at 13 hour: 19.071513414382935 %
Likelihood of crime at 14 hour: 21.921277046203613 %
Likelihood of crime at 15 hour: 24.97250735759735 %
Likelihood of crime at 16 hour: 27.56119668483734 %
Likelihood of crime at 17 hour: 30.3983211517334 %
Likelihood of crime at 18 hour: 32.50309526920318 6 %
Likelihood of crime at 19 hour: 32.97823965549469 %
Likelihood of crime at 20 hour: 33.582812547683716 %
Likelihood of crime at 21 hour: 34.2804700136184 7 %
Likelihood of crime at 22 hour: 33.93738865852356 %
Likelihood of crime at 23 hour: 33.29563140869140 6 %
```

Figure 18: Network 3: Prediction Output

D. Network 3.1 [20]

Network 3.1 was retrained to give the same output as Network 3 but the inputs to the network were changed to be the following:

- Year
- Month
- Day
- Hour
- Minute
- Latitude
- Longitude
- Distance from nearest Graffiti
- Distance from nearest Drinking Fountain

he following two plots show the accuracy and loss of our network throughout the training:

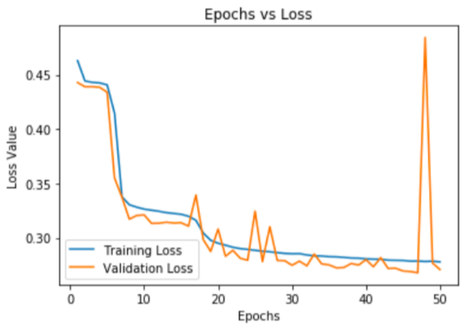


Figure 19: Network 3.1: Epoch vs Loss

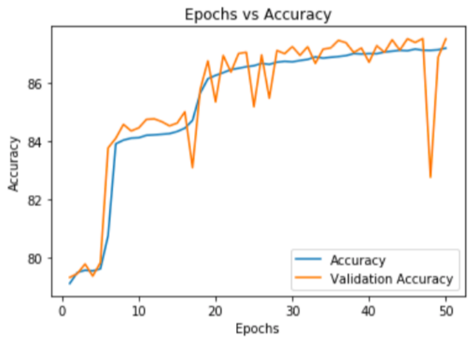


Figure 20: Network 3.1: Epoch vs Accuracy

Although this network has a test accuracy of 87.23%. The prediction output gives a near 100% chance of crime throughout the day/hour no matter what the input [20].

E. Network 3.2 [21]

Network 3.2 was retrained to give the same output as Network 3 but the inputs to the network were changed to be the following:

- Year
- Month
- Day
- Hour

- Minute
- Latitude
- Longitude
- Distance from nearest Graffiti
- Distance from nearest Drinking Fountain
- Google Trend Data

The following two plots show the accuracy and loss of our network throughout the training:

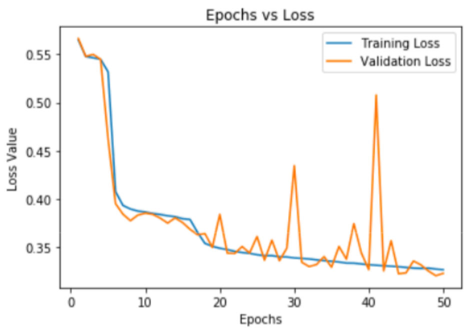


Figure 21: Network 3.2: Epoch vs Loss

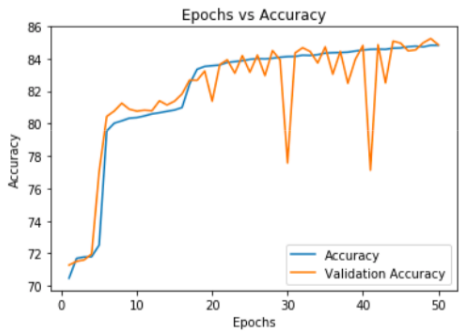


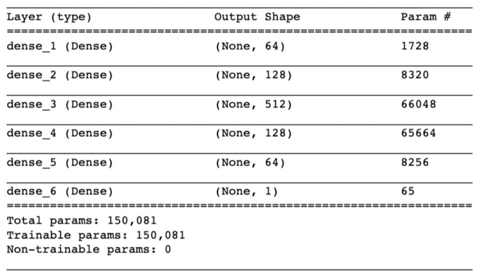
Figure 22: Network 3.2: Epoch vs Accuracy

Although this network has a test accuracy of 84.69%. The prediction output gives a better output than Network 3.1 which therefore means that the Google Trend data does have an effect on the network's ability to succeed. [21].

F. Network 4 [22]

Here we attempted to predict which neighborhoods would have crime given the year, month, day and hour. This was done in a similar style to a multi-class classification where each neighborhood was a class and the network was trained to output the multiple classes where a crime would happen for a give year, month and day.

We used the following network architecture which had the best performance:



```
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 64)                1728
_________________________________________________________________
dense_2 (Dense)              (None, 128)               8320
_________________________________________________________________
dense_3 (Dense)              (None, 512)               66048
_________________________________________________________________
dense_4 (Dense)              (None, 128)               65664
_________________________________________________________________
dense_5 (Dense)              (None, 64)                8256
_________________________________________________________________
dense_6 (Dense)              (None, 1)                 65
=================================================================
Total params: 150,081
Trainable params: 150,081
Non-trainable params: 0
_________________________________________________________________
```

Figure 23: Network 4 Architecture

The following two plots show the accuracy and loss of our network throughout the training:

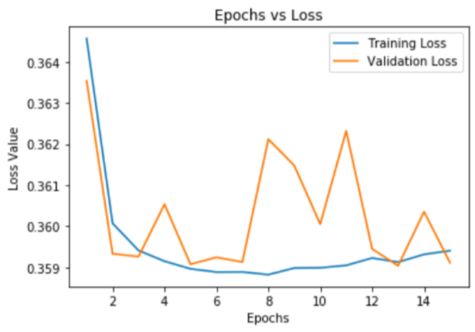


Figure 24: Network 4: Epoch vs Loss

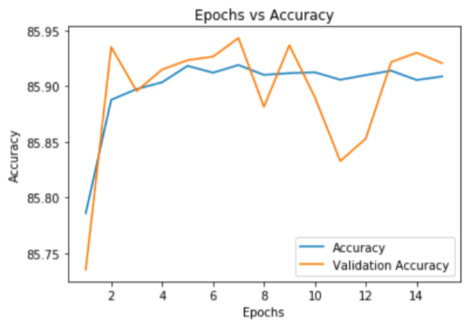


Figure 25: Network 4: Epoch vs Accuracy

On the test dataset we had an accuracy of 86.15%.

V. Conclusion

Predicting crime is no easy task due to its dependency on multiple factors, through our different experiments we were able to produce networks that found various different patterns in crime and used them to improve prediction results. We do believe that having a larger dataset could lead to further improvements in our network. Furthermore, certain factors like weather which we wanted to incorporate in our dataset were not available and using them alongside the other variables we introduced could lead to even better results.

We also believe that using multiple networks together is the way forward. Such as using network 2 and 4 to predict where crime will happen and what time of crime will occur.

References

[1] P. Carlen, *Women, Crime and Poverty*. Open University Press, dec 1988.

[2] A. M. Olligschlaeger, "Artificial neural networks and crime mapping."

[3] C. Yu, M. W. Ward, M. Morabito, and W. Ding, "Crime forecasting using data mining techniques," in *2011 IEEE 11th International Conference on Data Mining Workshops*, Dec 2011, pp. 779–786.

[4] A. Stec and D. Klabjan, "Forecasting crime with deep learning," *arXiv preprint arXiv:1806.01486*, 2018.

[5] "Vancouver open data catalogue." [Online]. Available: https://data.vancouver.ca/datacatalogue/

[6] "Vancouver crime data." [Online]. Available: https://data.vancouver.ca/datacatalogue/crime-data.htm

[7] "Vancouver local area boundary data." [Online]. Available: https://data.vancouver.ca/datacatalogue/localAreaBoundary.htm

[8] "Vancouver graffiti data." [Online]. Available: https://data.vancouver.ca/datacatalogue/graffitiSites.htm

[9] "Open data catalogue." [Online]. Available: https://data.vancouver.ca/datacatalogue/drinkingFountains.htm

[10] 2019. [Online]. Available: https://trends.google.com/trends/explore?geo=CA-BC&q=crime

[11] Y. Khaireddin, "Google trend adder - ele494-project." [Online]. Available: https://github.com/NasirKhalid24/ELE494-Project/blob/master/Scripts/googletrend.py

[12] N. Khalid and Y. Khaireddin, "Final crime dataset - ele494-project." [Online]. Available: https://github.com/NasirKhalid24/ELE494-Project/blob/master/Datasets/final_crime.csv

[13] N. Khalid, "Distance from graffiti calculator - ele494-project." [Online]. Available: https://github.com/NasirKhalid24/ELE494-Project/blob/master/Scripts/update_graffiti.py

[14] ——, "Distance from drinking fountain calculator - ele494-project." [Online]. Available: https://github.com/NasirKhalid24/ELE494-Project/blob/master/Scripts/update_drinkingfountain.py

[15] Y. Khaireddin, "Network 1 - ele494-project." [Online]. Available: https://github.com/NasirKhalid24/ELE494-Project/blob/master/Notebooks/Network_1.ipynb

[16] ——, "Bins dataset- ele494-project." [Online]. Available: https://github.com/NasirKhalid24/ELE494-Project/blob/master/Datasets/crime_ymdn_bins.csv

[17] ——, "Code to create bins dataset - ele494-project." [Online]. Available: https://github.com/NasirKhalid24/ELE494-Project/blob/master/Scripts/create_bins.py

[18] N. Khalid, "Network 2 - ele494-project." [Online]. Available: https://github.com/NasirKhalid24/ELE494-Project/blob/master/Notebooks/Network_2.ipynb

[19] ——, "Network 3 - ele494-project." [Online]. Available: https://github.com/NasirKhalid24/ELE494-Project/blob/master/Notebooks/Network_3.ipynb

[20] ——, "Network 3.1 - ele494-project." [Online]. Available: https://github.com/NasirKhalid24/ELE494-Project/blob/master/Notebooks/Network_3.1.ipynb

[21] Y. Khaireddin, "Network 3.2 - ele494-project." [Online]. Available: https://github.com/NasirKhalid24/ELE494-Project/blob/master/Notebooks/Network_3.2.ipynb

[22] N. Khalid, "Network 4 - ele494-project." [Online]. Available: https://github.com/NasirKhalid24/ELE494-Project/blob/master/Notebooks/Network_4.ipynb