Epidemics Graph Neural Network Link Prediction

Jaykumar Patel
patel.jay4802@utexas.edu

Afnan Mir
afnanmir@utexas.edu

Abstract

The COVID-19 pandemic showed that contact tracing helped mitigate the spread of the virus. However, manual contact tracing is slow and prone to inaccuracies. Thus, this project aims to automate contact tracing by utilizing Graph Neural Networks (GNNs) for link prediction. Our network analysis shows that the contact network is mostly exponential with hints of scale-free properties. Additionally, our analysis identifies a discernible pattern in mobility, characterized by an increase in travel during the weekends. Furthermore, we were able to achieve an AUC of 0.91 for static link prediction by using the GCN architecture and performing feature engineering. We then performed a temporal link prediction, which uses previous contact networks to predict future contacts. We achieved an ROC AUC score of 0.79 on nodes that were seen during training, and 0.54 on nodes that were not seen during training.

1. Introduction and Motivation

When COVID-19 first appeared, manual contact tracing was deployed to mitigate the initial outbreak. Contact tracing is the process of tracking how the virus spreads by identifying people who may have come in contact with an infected person, and then asking them to isolate and get tested.

However, the pandemic revealed that the COVID-19 virus spread faster than manual contact tracing [2]. Thus, this project's objective is to automate contact tracing by incorporating machine learning using GNNs to hopefully increase the mitigation of the spread of COVID-19 when compared to manual contact tracing. Firstly, we will create and analyze contact networks. Then we will use GNNs to perform static link prediction using a 5-Day Contact Network. Finally, we will perform temporal link prediction on the daily contact networks.

2. Previous Work

Mathematical models, classical machine learning models, and graph-based machine learning models have been used to predict virus spread.

The SEIRD model is a mathematical model that predicts the change in Susceptible, Exposed, Infected, Recovered, and Deceased people over time by using differential equations. [3]. The Susceptible-Infected-Recovered (SIR) model is a simpler version of the SEIRD model [13].

The Long Short Term Memory model is a machine learning model that has been used to predict the number of cases over time [3]. A hybrid of SIRD and LSTM helps account for time-dependent parameters of the SIRD model [1]. Also, GNNs, which are graph-based ML models, have been used on mobility data to predict virus spread and for link prediction for contact tracing [10][11].

Additionally, there have been studies done on performing graph learning on networks that are dynamic in nature, such as contact networks. The Temporal Graph Network (TGN) framework allows us to perform deep learning tasks on a sequence of graphs by using memory modules and graph-based operators [9].

We hope to build off of these previous works by utilizing GNNs and other deep learning techniques to be able to predict contacts between individuals. This is a non-trivial and novel task, as it requires the model to potentially learn a graph's structure based on its structure at previous time steps as well as node features.

3. Approach

3.1 Network Generation and Simulation

We used the Foursquare dataset from the months of July and August 2020 to build contact networks of Austin, TX [8]. Each entry contains a device ID, a location ID, a UTC date and hour, and a dwell time, which tell us when and how long a person visited a location. Given this data, we generated a contact graph of Austin.

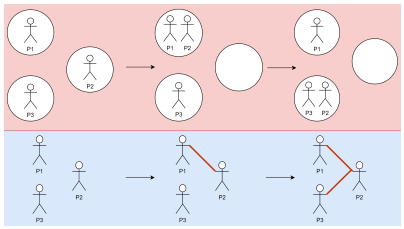

Figure 1: Example of contact network generation

Firstly, we used data from July 1st, 2020 to July 5th, 2020 to create a sample contact network. Our nodes were all the unique device IDs in the dataset, which correspond to people. For our edges, we used the following logic: we ignored entries with a dwell time of less than 60 minutes, as we assumed this was not enough time to make significant contact with others. Then, we used the UTC date and hour with the dwell time to determine the arrival and departure time interval for each entry. We then compared every entry with every other entry. If the entries' locations were the same and if intervals overlapped by at least 60 minutes, we considered this as a contact between the two people and added an edge between them. We will call this the 5-Day Contact Network. This network captures the meaningful contacts that occurred between people from July 1st, 2020 to July 5th, 2020. Figure 1 shows an example of a sample contact network generation.

Then we also created a set of 62 contact networks, one for each day from July 1st, 2020 to August 31st, 2020. We used the same logic as before to create these individual contact networks. If two people were at the same location at the same time for at least 60 minutes, we created an edge between them. We will call this set of networks the Temporal Contact Networks. Each day's network captures the meaningful contacts that occurred between people for that day. Furthermore, we analyzed how the clustering coefficient and average node degree changed over time by calculating these metrics for each network in the Temporal Contact Networks.

Furthermore, we created an SIR simulation using the Temporal Contact Networks, which we ran from July 1st, 2020 to August 31st, 2020. Here are the parameters and assumptions that were made for the simulation:

- Contact between people that is less than 60 minutes is not considered significant enough to spread the virus.
- If a susceptible person comes into contact with an infected person for at least one hour, then they get infected with a probability of 0.30. This is called the infection rate (IR) and it is constant.
- An infected person will recover after seven days. This is called the recovery period (RP) and it is constant.
- A person can only be infected if they were previously susceptible, and a person can only be recovered if they were previously infected.
- Initially, 20% of the people, chosen at random, are infected. The rest are susceptible.
- Only infected people can infect others.

The simulation algorithm is shown in Algorithm 1. The simulation was run on each network in the Temporal Contact Networks. The simulation results were analyzed to see how the virus spread over time. This approach is scaleable, as the simulation can be run on any number of Temporal Contact Networks. Thus, we can use this approach to simulate the spread of the virus over a longer period of time.

Algorithm 1 SIR Simulation

Input: Temporal Contact Networks (TCN: Array of Contact Networks), IR, RP
Initialize: Set 20% nodes with $state = I$, Set 80% nodes with $state = S$, Set all nodes with $time\_of\_recovery = \infty$
for i = 1 to len(TCN) **do**
  for each node n where $n.state = I$ **do**
    for each neighbor m of n in TCN[i] where $m.state = S$ **do**
      if $rand(0, 1) <=$ IR **then**
        $m.state = I$
        $m.time\_of\_recovery = i + RP$
      end if
    end for
    if $i = n.time\_of\_recovery$ **then**
      $n.state = R$
    end if
  end for
end for

3.2 Machine Learning

After generating and analyzing the 5-Day Contact Network, and Temporal Contact Networks, and performing the SIR simulation, we moved towards leveraging graph learning techniques to perform link prediction, which is the fundamental task behind automating contact tracing. Initially, we focused on performing link prediction on a static graph. In order to have enough data to train and evaluate our models, we used the 5-Day Contact Network.

3.2.1 Static Link Prediction

Our first goal was to create a baseline link prediction model. We used the node2vec algorithm to generate node embeddings for each node in the graph [4]. We then generated the dataset of edges. To perform link prediction, we need a set of positive edges, which are the edges present in the network, and we need a set of negative edges, which are the edges not present in the network. This allows us to boil down the link prediction to a binary classification problem. Given our network, we created a set of negative edges that was equal in size to the set of positive edges to ensure balanced training. Using the node2vec embeddings and the set

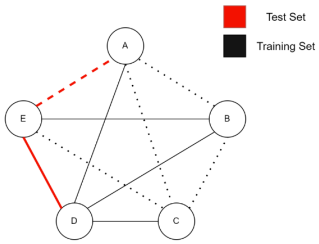

Figure 2: Example of positive and negative edges for train and test data

of positive and negative edges, we trained a GraphSAGE model to perform link prediction on the static 5-Day Contact Network [5]. Our training set consisted of all positive and negative edges from the 5-Day contact network, and our testing set consisted of all new positive and negative edges found on the sixth day.

A sample example of training and testing data is shown in Figure 2. The black lines represent training data and the red lines represent testing data. The solid lines represent positive edges and dotted lines represent negative edges. The model is trained on the black lines, and attempts to predict the red lines. This allows us to boil down the link prediction to a binary classification problem.

After creating the baseline model, we searched for ways to improve the model's performance on the graph. This would include performing feature engineering techniques to add dimensions to our node embeddings and exploring the use of other GNN architectures such as the Graph Convolutional Network (GCN) and/or the Graph Attention Network (GAT) [7] [12]. We hoped to be able to finetune the model and improve its performance to the point we could use it to perform link prediction on the Temporal Contact Networks. This approach is scalable because generating the training and testing edge sample, node embeddings, and node features can be done on a network of any size; however, generating the node2vec embeddings for a large network can be computationally expensive.

3.2.2 Temporal Link Prediction

After performing static link prediction on the 5-Day Contact Network, we moved towards performing temporal link prediction on the Temporal Contact Networks.

Firstly, the Temporal Contact Networks are converted into a list of edge data entries, which we will call Temporal Edge Data (TED). Each data entry consists of the endpoints of the edge and the timestamp of when the edge existed. For example, if an edge between node A and B exists in the contact networks for July 1st, 2020 and July 2nd, 2020, but does not exist in the contact network for July 3rd, 2020, then the edge data entries would be [(A, B, 1), (A, B, 2)]. Note how (A, B, 3) is not included in the list of edge data entries. This is because the edge between A and B does not exist in the contact network for July 3rd, 2020. Thus, the TED will only have positive edges. Once the complete TED is created, it is sorted by the timestamp. Then the first 85% of the TED is used as the training set, and the last 15% of the list is used as the testing set. Finally, the TED gets split into batches, which get processed sequentially by the model.

In order to perform temporal link prediciton, we utilized the TGN framework [9]. The TGN framework allows us to perform deep learning tasks on a sequence of graphs. The TGN framework consists of 5 main modules: memory, message function, message aggregator, memory updater, and embedding.

The memory module is a memory bank that stores the temporal information of the nodes in the graph. The message function module takes in a batch of edge data from the TED, and generates two messages for each edge, one for each endpoint. If a node is an endpoint for multiple edges, then it will have multiple messages. The original paper utilizes "identity" as the message function, which essentially concatenates the inputs. In contrast, our model uses a Multi-Layer Perceptron (MLP) as the message function, which adds a learnable parameter improving the complexity of the model. The message aggregator combines the raw messages from the message function by taking the latest message for each node. Thus, the output of the message aggregator is one message for each node. The memory updater is a Recurrent Neural Network (RNN) that takes the messages from the message aggregator to updates the nodes' temoral memory. The embedding module utilizes the Temporal Graph Attention (TGAT) model to generate node embeddings given a node and its neighbors' memories from the previous batch. Node embeddings for two nodes are passed through an MLP to generate a probability of the two nodes being connected. The model is trained using the Adam optimizer and the binary cross entropy loss function. Additionally, the original model uses no node features. To improve performance, we generate and utilize static node features identical to the features used for static link prediction. The model's architecture is shown in Figure 3.

4. Experimental Setup and Results

4.1 5-Day Network Analysis

In Figure 4, we can see the 5-Day Contact Network. Network properties for this network were calculated. The average node degree is 102.599, the network diameter is 7, the average clustering coefficient is 0.627, and the average path length is 2.849. In addition to this, the degree distribution was mainly an exponential distribution with subtle hints of power-law as shown in Figure 5. This can be seen from the

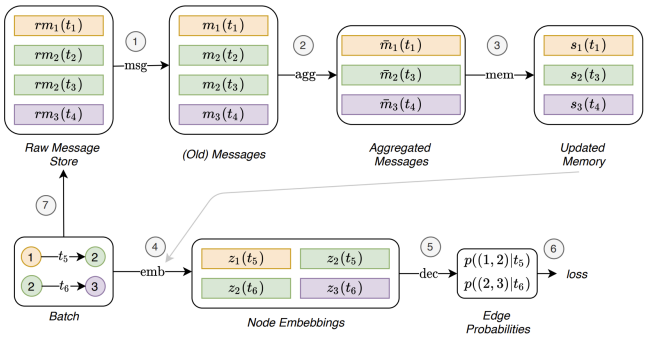

Figure 3: TGN Modules Architecture [9]

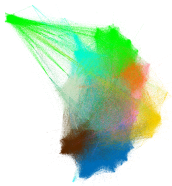

Figure 4: 5-Day Contact Network from July 1st, 2020 to July 5th, 2020

network itself, as we can see the presence of a few hubs in the network.

4.2 Temporal Network Analysis and Simulation

In addition, the simulation results are shown in Figure 6. The simulation results show that the number of infected people initially increases, but then decreases significantly and approaches zero. However, the number of recovered people exhibits the opposite behavior. This makes sense since the people who started with the infection at the beginning of the simulation recover after seven days; thus, they can no longer infect others nor become susceptible again. Also, the number of susceptible people decreases over time, which makes sense since they are getting infected and recovering. The maximum number of infected people on any given day is about 11,500 and the total number of people

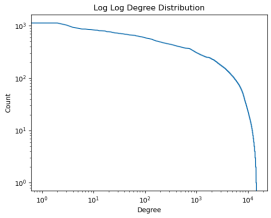

Figure 5: 5-Day Contact Network Degree Distribution

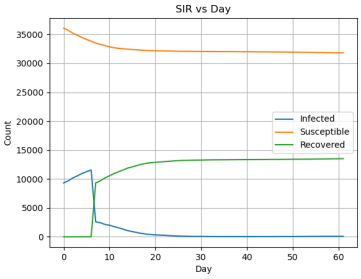

Figure 6: SIR Simulation Results

infected over the simulation is about 14,300.

It is important to note that this simulation was run on limited data. The total number of nodes in the simulation, which is the total number of people, is around 45,000, whereas the actual population in the Austin metropolitan area in 2020 was around 2 million. Furthermore, this model assumes a closed population, which is not the case in Austin, TX. People are constantly moving in and out of the city. Thus, the simulation results are not representative of the actual spread of the virus in Austin, TX. Having access to more data would allow the simulation to provide realistic results. However, this simulation does show that the virus can spread quickly, as seen in the first week of the simulation. Thus, it is important to have an efficient and accurate contact tracing system to mitigate the spread of the virus.

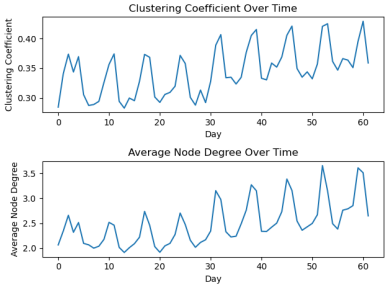

Figure 7: Clustering Coefficient and Average Node Degree for Temporal Contact Networks

We also analyzed the Temporal Contact Networks. The clustering coefficient and the average node degree for these networks are shown in Figure 7. The results reveal a seven-day periodic pattern with peaks occurring on the weekends. These results make sense since people are more likely to go out and socialize on the weekends. This increase in socialization on the weekends leads to more contacts, which leads to a higher clustering coefficient and average node degree.

4.3 Static Link Prediction

After performing the analysis on the 5-Day Contact Network and Temporal Contact Networks, we moved towards performing link prediction on the 5-Day Contact Network. For our baseline model, as stated in the approach, we used a GraphSAGE model with node2vec embeddings. We trained the model for 1000 epochs with the Adam optimizer, where the train set is the set of positive and negative edges from the 5-Day Contact Network, and the test set is the set of positive and negative edges on the sixth day [6]. The model achieved an AUC score of 0.49 on the test data, which is worse than random guessing. Part of this could be because no internal node features were used in the node embeddings.

To improve upon this, we performed feature engineering to generate additional features for each node. The features we added included average number of locations traveled to per day, average distance traveled per day, age, gender, and State-Age-Gender (SAG) score. By adding these features to the node2vec embeddings, we were able to improve the AUC score to 0.63, which is a significant improvement over the baseline model. We can see that the additional features helped the model learn the graph structure better.

We noticed that the features were of different scales, so we hypothesized that normalizing the features would improve the model's accuracy. This adjustment improved the performance of the model, leading to an AUC of 0.75.

Seeking further enhancement, we explored a different model architecture, specifically the GCN model. Utilizing the same features and data, the GCN model achieved an AUC score of 0.91, demonstrating a significant improvement over the initial model.

Furthermore, minimizing the false-negative rate is important. False negatives are instances where the model predicts a negative edge between two nodes, when in reality there is a positive edge between the two nodes. Recall is a measurement of the true positive rate and helps minimize the false negatives. It is evident that the recall for static link prediction is 0.83, which is derived from the confusion matrix. The final results are shown in Figure 8.

While these performance metrics establish the feasibility of link prediction on contact networks, it's crucial to note a limitation in the current approach—namely, its inability to capture temporal data.

4.4 Temporal Link Prediction

After performing static link prediction on the 5-Day Contact Network, we performed temporal link prediction on the Temporal Contact Networks. We utilized the TGN framework to perform the link prediction. Furthermore, we used the same node features as static link prediction. The

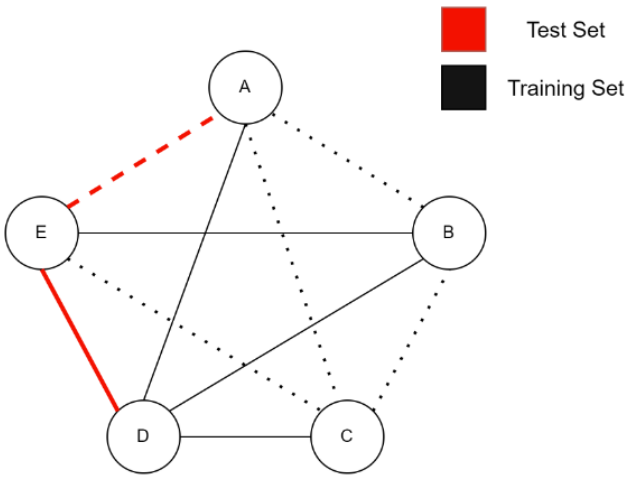

Figure 8: Static Link Prediction Results

model was trained such that it terminated after 5 consecutive epochs of no improvement in the validation loss.

The model achieved an ROC AUC score of 0.79 on nodes that were seen during training, and 0.54 on nodes that were not seen during training. This shows that the model was able to learn the graph structure and perform link prediction on the Temporal Contact Networks. However, the model's poor performance on unseen nodes suggests that the model works best on a closed population. Furthermore, it is evident that the recall for temporal link prediction is 0.78, which is derived from the confusion matrix. The final results are shown in Figure 9.

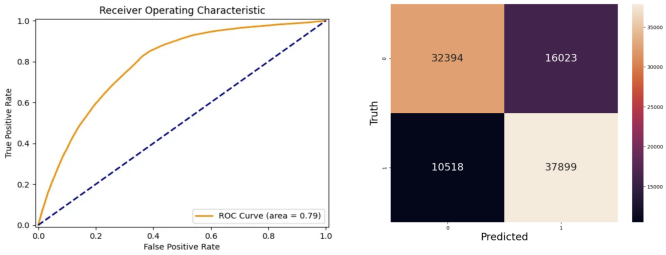

Figure 9: Temporal Link Prediction Results

5. Conclusion and Future Work

Through the analysis of the 5-Day Contact Network, we were able to determine that the network is mostly exponential in degree distribution, with hints of scale-free properties. Some people came into contact with many other people whereas others stayed within their cliques. The simulation showed that the number of infected people initially increases, but after a few days, decreases significantly.

The analysis of the Temporal Contact Networks highlights a pattern in mobility. Particularly, people tend to socialize more on the weekends, which leads to a higher clustering coefficient and average node degree.

Then, we were able to perform static link prediction on the 5-Day Contact Network. We were able to improve the model's performance by performing feature engineering and using the GCN model.

Finally, we were able to perform temporal link prediction on the Temporal Contact Networks using the TGN framework, achieveing a performance that is significantly better than random guessing for a closed population. These results are important because they show that link prediction on contact networks is a feasible task. Predicting contact between people can be crucial in mitigating the spread of the virus because it can lead to proactive measures such as quarantining and testing. For example, if person B is infected and the model predicts that person A will come into contact with person B, then person A can be mindful of their contact with person B and take proactive measures such staying home and getting tested. This can help mitigate the spread of the virus. However, the model's poor performance on unseen nodes suggests that the model works best on a closed population. Thus, we plan to improve the model's performance on unseen nodes for future work.

For future work, we propose adding edge features to the dataset, such as how long two people were in contact for and where the contact occurred. We also propose implementing dynamic node features, which are features of nodes that change over time. For example, we can add a feature that represents the number of people a node has come into contact with over the past week. We hypothesize that adding these features will improve the performance of temporal link prediction.

6 Contributions and Lessons Learned

For this project, Afnan generated and analyzed the 5-Day Contact Network and performed static link prediction. Jaykumar created and analyzed the Temporal Contact Networks, ran the SIR simulation, and performed temporal link prediction.

From this project, we learned a great deal about utilizing network theory and GNNs in the space of epidemeology. First, we learned about how to leverage mobility datasets to generate contact networks between individuals within a metropolitan area and how to simulate the spread of a virus through a population. Through this process, we also learned of some pitfalls of using mobility datasets, specifically Foursquare, in the simulation of disease spread. With our simulation, we did not achieve a similar trend to the actual spread of COVID-19 in Austin, TX. This is mainly due to the fact that the dataset does not include a large enough sample of the population to accurately simulate the spread and does not account for people moving in and out of the city.

We also learned about the effectiveness of GNNs in detecting patterns and performing predictions on structured data such as contact networks. We were able to achieve impressive results in the task of link prediction on the 5-Day Contact Network with a simple GCN model with some feature engineering. We also were able to achieve promising results in the very challenging task of dynamic link prediction on the Temporal Contact Networks. These promising results without performing heavy feature engineering, using large scale models, or performing any other potential optimizations show that the potential of GNNs in the space of epidemiology is immense.

References

[1] A. Bousquet, W. H. Conrad, S. O. Sadat, N. Vardanyan, and Y. Hong. Deep learning forecasting using time-varying parameters of the sird model for covid-19, February 22 2022.

[2] S. Flaxman, S. Mishra, A. Gandy, H. J. T. Unwin, T. A. Mellan, H. Coupland, and et al. Estimating the number of infections and the impact of non-pharmaceutical interventions on covid-19 in european countries: technical description update, 2020.

[3] T. Geroski, A. Blagojevic, D. M. Cvetković, A. M. Cvetković, I. Lorencin, S. B. Šegota, D. Milovanovic, D. Baskic, Z. Car, and N. Filipovic. Epidemiological predictive modeling of covid-19 infection: Development, testing, and implementation on the population of the benelux union, October 28 2021.

[4] A. Grover and J. Leskovec. node2vec: Scalable feature learning for networks, 2016.

[5] W. L. Hamilton, R. Ying, and J. Leskovec. Inductive representation learning on large graphs, 2018.

[6] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2017.

[7] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks, 2017.

[8] C. D. Lab. Foursquare Community Mobility Data with Basemap (US), 2020.

[9] E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, and M. Bronstein. Temporal graph networks for deep learning on dynamic graphs, 2020.

[10] K. Skianis, G. Nikolentzos, B. Gallix, R. Thiebaut, and G. Exarchakis. Predicting covid-19 positivity and hospitalization with multi-scale graph neural networks, March 31 2023.

[11] C. W. Tan, P.-D. Yu, S. Chen, and H. V. Poor. Deeptrace: Learning to optimize contact tracing in epidemic networks with graph neural networks, 2023.

[12] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks, 2018.

[13] R. S. Yadav. Mathematical modeling and simulation of sir model for covid-2019 epidemic outbreak: A case study of india, May 21 2020.