Spatio-temporal covid-19 prediction using hybrid Gated Attention Network-RNN model

Project submitted
In partial fulfilments of the requirements for the degree of

MASTER OF COMPUTER APPLICATION

By

TANMAY DAS
Roll No: **001910503021**
Examination Roll No.: **MCA226021**
Registration No: **149884 of 2019-20**

Under the supervision of

DR. ANASUA SARKAR

Department of Computer Science & Engineering
Faculty of Engineering and Technology
Jadavpur University
Kolkata – 7000 032
India

Jadavpur University

Faculty of Engineering and Technology
Department of Computer Science & Engineering

CERTIFICATE

This is to clarify that the project entitled “**Spatio-temporal covid-19 prediction using hybrid Gated Attention Network-RNN model**” has been completed by Tanmay Das. This work is carried out under the supervision of Dr. Anasua Sarkar in partial fulfilment for the award of the degree of **Master of Computer Application** of the department of **Computer Science and Engineering**, Jadavpur University, during the session 2019-2022. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said degree.

Dr. Anasua Sarkar
Project supervisor
Computer Science & Engineering
Jadavpur University

Countersigned:

Prof. Anupam Sinha
Head of the department
Computer Science & Engineering
Jadavpur University

Prof. Chandan Mazumdar
Dean
Faculty of Engineering & Technology
Jadavpur University

Jadavpur University

Faculty of Engineering and Technology
Department of Computer Science & Engineering

CERTIFICATE

This is to certify that the project entitled **“Spatio-temporal covid-19 prediction using hybrid Gated Attention Network-RNN model”** has been submitted by **Tanmay Das** in partial fulfilment of the requirements for the award of the degree of **Master of Computer Application** in the department of **Computer Science & Engineering**, Jadavpur University, during the period 2019-2022 has been carried out under my supervision and that this work has not been submitted elsewhere for obtaining a degree.

EXAMINER:

----------------------------------- **-------------------------------------**

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION OF ORIGINALITY AND COMPLIANCE OF ACADEMIC ETHICS

I hereby declare that this project contains original work by the undersigned candidate, as part of his Master of Computer Application (MCA) studies.

All information in this document have been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by these rules and conduct, I have fully cited and referenced all material results that are not original to this work.

Name: Tanmay Das
Roll No: 001910503021
Examination Roll No.: MCA226021
Project Title: Spatio-temporal covid-19 prediction using hybrid Gated Attention Network-RNN model.

Signature with date

ACKNOWLEDGEMENTS

On the submission of "Spatio-temporal covid-19 prediction using hybrid Gated Attention Network-RNN model.", I would like to convey my sincere gratitude to Dr. Anasua Sarkar, Assistant Professor, Department of Computer Science & Engineering, Jadavpur University for her valuable suggestions throughout the duration of the thesis work. I am really grateful to her for constant support which immensely helped me to fully involve myself in this work and develop new approaches in the field of spatio-temporal modelling.

Lastly, I would like to thank all my teachers, classmates, guardians and well-wishers for encouraging and co-operating me throughout the development of this thesis.

Regards,

TANMAY DAS

Roll Number: 001910503021
Examination Roll No.: MCA226021
Registration Number: 149884 of 2019-20

Signature with date

ABSTRACT

As we all know covid19 pandemic heated us deeply. The entire world goes down in lockdown many times. Many major countries' economies collapsed, and the healthcare system collapsed due to the major number of covid 19 patients. However, many countries or states are proposed an international travel ban in early 2020, so there are many reasons to transmissions would to occur due to domestic travel in-between states or cities. Different states or cities have different populations, some cities have dense populations and some have fewer populations. While extensive research has analyzed the issue on a large scale to predict the pandemic, here we focus on entire countries and states, mainly comprising states and cities, for a more accurate prediction of the number of cases (like new cases, death cases, etc.) we use different country or state claim data, demographic and geographic proximity between location and divide them based on populations, and integrate pandemic dynamic into the deep learning model. We perform an attention network model to predict the pandemic data. It uses a graph attention network to dynamically predict cases for a certain period in a country or state. By using real-world data and case counts, this model accurately predicts covid19 status and helps us to prevent pandemics in the future.

Keywords: *Graph Attention Network, RNN, Pandemic Prediction, Deep learning*

CONTENTS

1 INTRODUCTION

1.1 BACKGROUND

The SARS-CoV-2 virus causes Coronavirus Illness (COVID-19), which is an infectious disease. It was initially appeared in late 2019 in Wuhan, China, but it has since spread to countries all over the world. As of April 2022, the illness had a global effect of roughly 528 million cases and 63 million deaths. COVID-19 [1] is a highly contagious virus that spreads quicker than influenza but not as quickly as measles, one of the most common human illnesses.

To combat the disease's fast spread, practically every country on the planet has implemented a variety of World Health Organization (WHO) recommended prevention measures. COVID-19 has been shown to spread mostly through human contact between persons who are within 6 feet of one another. As a result, the two most important defences are social separation and mask wear, both of which attempt to limit interpersonal interactions. This type of transmission is caused by coughing, sneezing, talking, and other types of contact with an infected person's respiratory droplets. COVID-19's effects must be managed and limited in society due to the ease with which this potentially lethal disease can be transmitted. As a result, the capacity to predict the spread of COVID-19 becomes important in the world's ability to develop a sufficient response to the pandemic.

1.2 MOTIVATION

Pandemic infections, such as the new coronavirus disease (COVID19), are quickly spreading over the world, posing a serious danger to global public health [23]. As a result, being able to forecast pandemic breakouts early and correctly is crucial for helping to establish suitable policies and reduce losses.

To anticipate the COVID-19 pandemic, a variety of epidemiological (e.g., SIR, SEIR) and deep learning models (e.g., LSTM) have been used. They are, however, confronted with three significant obstacles: 1) They normally create a different model for each place (for example, one for each county), ignoring physical closeness and relationships with other regions. Or the projections are based only on observable trends from other areas, with no direct modelling of interregional relationships. In reality, due to population migrations or demographic similarities, a location's disease patterns are generally comparable to surrounding regions or demographically similar locations. 2) The majority of existing models are based on COVID-19 case report data. There is a lot of underreporting in these statistics, as well as other data quality concerns. 3) SIR and SEIR models are deterministic epidemiological models, to fit the complete curve of illness counts, they apply a set of differential equations. Because these models are based on a small number of variables, they are unable to represent complicated short-term patterns like superinfection or time-varying infectivity. Deep learning-based models, on the other hand, can only anticipate known data patterns and can only make correct predictions in a limited amount of time. As a result, while there are methodologies that allow for short- or long-term prediction models of disease outbreaks, current models do not give reliable models spanning both time frames.

1.3 LITERATURE SURVEY

Prior to starting the research, it is required to evaluate past works in the field to have a baseline understanding of the current level of knowledge on the issue. This literature review summarizes the most recent research on COVID-19 time-series forecasting. Traditional epidemic prediction models, such as the Susceptible-Infected-Removed (SIR) [2] model and Susceptible-Exposed-Infected-Removed (SEIR) [3] models and their derivatives, employ compartmental-based models to estimate disease transmission patterns at the population level. Time series learning methodologies [14], such as curve-fitting or autoregression [4], are also used in certain studies to forecast pandemics. Deep learning techniques [15] have been created to portray epidemic or pandemic modelling as time series prediction issues, in addition to classic statistical models.

Epidemiological models are often used in disease research to examine infectious disease development patterns and estimate epidemic implications [5]. For example, SIR model estimates the number of persons infected with a disease in a closed society over time. In the early stages of the pandemic, the numbers of susceptible (S), infected (I), and recovered (R) persons are used to anticipate the spread of COVID-19 in China, as well as other nations including South Korea, India, Australia, the United States, and Italy. Other studies have used the SEIR paradigm to extend this model even further. This model is similar to SIR, except it additionally takes into account the exposed (E) population. In nations like China and Italy, the SEIR model has also been used to anticipate COVID-19 spread. The SEIR model has also been used to assess the impact of contact tracing, testing, and containment techniques in the presence of infection hotspots. Overall, the researchers were able to provide encouraging findings and projections for the countries studied.

For influenza-like disease incidence predictions, several studies blend deep neural networks (DNN) [6] with causal models. Deng et al [16] suggested a graph message passing framework to represent illness spread over time by combining learnt feature embeddings and an attention matrix. Yang et al [3] and his colleagues used past pandemic data to pretrain the LSTM, which they then used to forecast the course of COVID-19 in China. For COVID-19 prediction, Kapoor et al [17] used a basic graph neural network (GNN). However, instead of long-term evolution, these models simply anticipate the following day. Deep learning-based models still have a long way to go in terms of long-term prediction accuracy. Furthermore, DNN-based approaches have a critical flaw: they can only forecast recognised patterns from input data without comprehending the long-term trend [7]. For example, if all case numbers are growing early in the pandemic, these models are unlikely to forecast a downward trend in the future. As a result, DNN models have a hard time making long-term predictions.

1.4 OBJECTIVE

I propose a hybrid gated attention-RNN model for pandemic prediction based on real-world evidence like claims data and COVID-19 case surveillance data in this paper. We map physical areas (such as a county or a state) to nodes on a graph and create edges based on geographic proximity and demographic similarity. Each node has a collection of static and dynamic characteristics. To capture interactions of comparable sites, we use graph attention network (GAT) [18]. Then, using transmission dynamics of epidemiological models, I anticipate the number of infected individuals for a set period in the future while also placing physical limits on projections. I use hybrid gated attention-RNN to forecast the number of new cases at the state and county levels.

2 METHODOLOGY

2.1 NEURAL NETWORKS

Neural networks, which try to generate a map of inputs and outputs similar to how the brain reacts to numerous sensory inputs, are scientifically inspired by neurons in the brain. A sort of computer network is a neural network. By combining forward and backward passes over, this mapping may be learnt. There are a number of input-output pairings to choose from. Feedforward neural networks (FNN) [9][10] are a significant component in this thesis, as are deep learning models like the ones used in this study. The FNN attempts to improve a function f that transfers an input x to an output y. It accomplishes this by forwarding data via all of the inputs until it reaches y, hence the name. The following formula may be used to represent a typical FNN:

$$f(x) = f_1\left(f_2\left(\dots\dots f_{n-1}(f_n)\right)\right) \qquad (1)$$

where f_i is the ith layer in the neural network containing n layers. Each layer in the neural network contains a set of nodes that takes in the output of previous layers as its input. These outputs are then multiplied by a series of weights, which the network will attempt to learn in approximating the target function.

2.1.1 TIME-SERIES FORECASTING MODELS

The key distinction between time-series forecasting and other machine learning applications is the type of data used [12]. When it comes to producing predictions, most machine learning models give prior observations roughly equal weights, while time-series forecasting adds an explicit order dependence between data as a crucial characteristic. In the forecasting process, models are fitted to existing data and used to anticipate future observations. All estimates are based on what has previously happened because the model has no means of predicting what will happen in the future. Time-series forecasting has several applications in illness tracking and is the foundation of our machine learning algorithms, whose histories are explained below.

2.1.2 RECURRENT NEURAL NETWORK (RNN)

A recurrent neural network is a feed-forward neural network with an internal memory that is an extension of the feed-forward neural network. RNN [11][12] is recurrent in nature since it executes the same function for each data input, and the current input's outcome is dependent on the previous calculation. The output is replicated and transmitted back into the recurrent network when it is created. It evaluates the current input as well as the output it has learnt from the prior input when making a decision.

RNNs, unlike feed-forward neural networks, may process sequences of inputs using their internal state (memory). As a result, activities like unsegmented, linked handwriting recognition or speech recognition are possible. All of the inputs in other neural networks are independent of one another. In an RNN, however, all of the inputs are connected to one another.

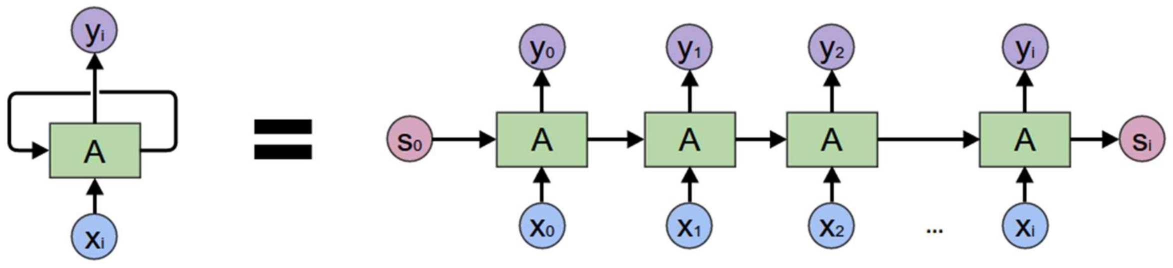


Figure 1: visualization of RNN cell [24]

2.1.3 GRAPH NEURAL NETWORK (GNN)

Data in a graph structure differs from data in a matrix or vector form. When each cell in a matrix or vector structure is regarded to be a cell, altering the sequence of these cells compromises the data's integrity. Graph structured data, on the other hand, is isomorphic. The graph's nodes, as well as the linkages between them, contain a wealth of information. GNNs [13][7] are neural networks that are designed to learn from graph-structured input.

2.1.4 Graph Attention Network (GAT)

The graph attention network [18] combines a graph neural network and an attention layer, as the name implies. To comprehend graph attention networks, we must first comprehend what an attention layer is and what graph-neural networks are. As a result, this section can be separated into two parts. We'll start with a fundamental grasp of the graph neural network and the attention layer, then move on to the combination of the two.

Attention layer divide in four different parts:

1) **Simple linear transformation**: In order to obtain sufficient expressive power to transform the input features into higher level features, at least one learnable linear transformation is required.

$$z_i^{(l)} = W^{(l)} h_i^{(l)} \tag{2}$$

2) **Attention Coefficients**: We then compute a pair-wise un-normalized attention score between two neighbors [16].

$$e_{ij}^{(l)} = \mathrm{LeakyReLU}(\vec{a}^{(l)^T}(z_i^{(l)} || z_j^{(l)})) \tag{3}$$

3) **SoftMax**: This makes coefficients easily comparable across different nodes, we normalize them across all choices of ***ij*** using the SoftMax function

$$\alpha_{ij}^{(l)} = \frac{\exp(e_{ij}^{(l)})}{\sum_{k \in \mathcal{N}(i)} \exp(e_{ik}^{(l)})} \tag{4}$$

4) **Aggregation**: This step is similar to GCN. The embeddings from neighbors are aggregated together, scaled by the attention scores [17].

$$h_i^{(l+1)} = \sigma\left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(l)} z_j^{(l)}\right) \tag{5}$$

2.1.5 MULTI-HEAD ATTENTION MECHANISM

The multi-head attention method allows the model to learn an attention coefficient by traversing various representation subspaces. Multi-head attention mechanism techniques are commonly used to make the self-attention [19] learning process more robust. Take, for example, the chosen multi-head attention mechanism [17]: K independent attention mechanisms conduct the aforementioned transformation across K heads (i.e., K independent attention processes), and their resultant features are concatenated together to generate an output feature representation. After that, the final result is created by averaging the feature representation concatenation. This procedure is explicitly defined as follows:

$$\begin{cases} h_i^{l+1} = \overset{K}{\underset{K=1}{\Vert}} \sigma\left(\sum_{j\in \mathrm{N}(i)} \alpha_{ij}^{K} \phi^{K} h_j^{l}\right), & \text{Concatenation} \\ h_i^{l+1} = \sigma\left(\frac{1}{K}\sum_{K=1}^{K}\sum_{j\in \mathrm{N}(i)} \alpha_{ij}^{K} \phi^{K} h_j^{l}\right), & \text{Averaging,} \end{cases} \tag{6}$$

2.2 ACTIVATION FUNCTION

To assess whether a neuron should be activated or not, the activation function builds a weighted sum and then adds bias to it. The weight, bias, and activation function of neurons in a neural network have been shown to influence how they perform. We would alter the weights and biases of the neurons in a neural network based on the output inaccuracy. Back-propagation is the term for this procedure. Because the gradients and error are provided at the same time to update the weights and biases, activation functions allow back-propagation. A neural network is simply a linear regression model that lacks an activation function. Non-linearity is dealt with using the activation function.

2.2.1 TANH ACTIVATION FUNCTION

The TanH activation function is given by

$$f(x) = \tanh(x) = (e^{2x} - 1)/(e^{2x} + 1) \tag{7}$$

The output of this function is normalised with respect to the input and can have values ranging from (-1, 1). Tanh is an excellent candidate for backpropagation because of the features listed above.

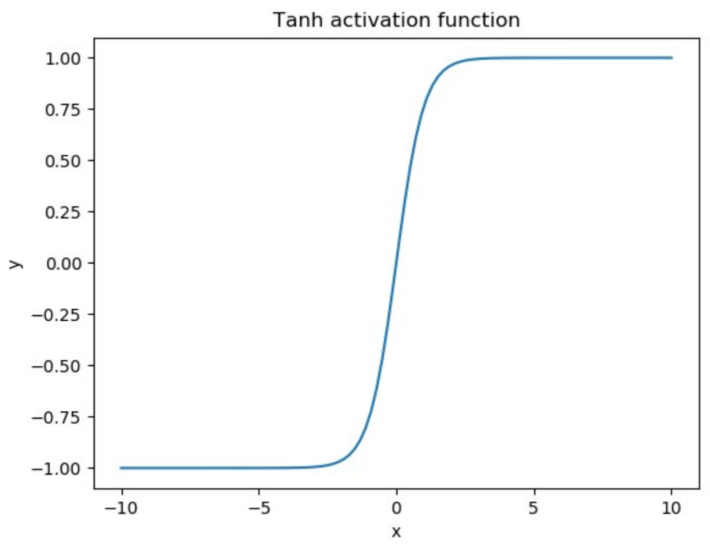


Figure 2: TanH Activation Function [25]

The graph is extremely close to another common alternative, the sigmoid activation function (S-shaped). As you can see from the graph, tanh is quite good at correlating inputs and outputs. Inputs that are substantially positive are normalised and mapped closer to 1, whereas inputs that are strongly negative are mapped closer to -1.

2.2.2 Leakey ReLU

The upgraded form of the ReLU activation function is the Leaky ReLU function. The gradient of the ReLU activation function is 0 for all input values less than zero, which would deactivate the neurons in that area and perhaps create the dying ReLU issue.

The term "leaky ReLU" was developed to describe a solution to this issue. We specify the ReLU activation function as a very tiny linear component of x instead of declaring it as 0 for negative values of inputs(x). This activation function's formula is as follows:

$$f(x) = \max(0.01 * x, x) \qquad (8)$$

If the input is positive, this method returns x; however, if the input is negative, it returns 0.01 times x. As a result, it also produces results for negative numbers. The gradient of the left side of the graph now has a non-zero value as a result of this simple alteration. As a result, no more dead neurons would be found in that area.

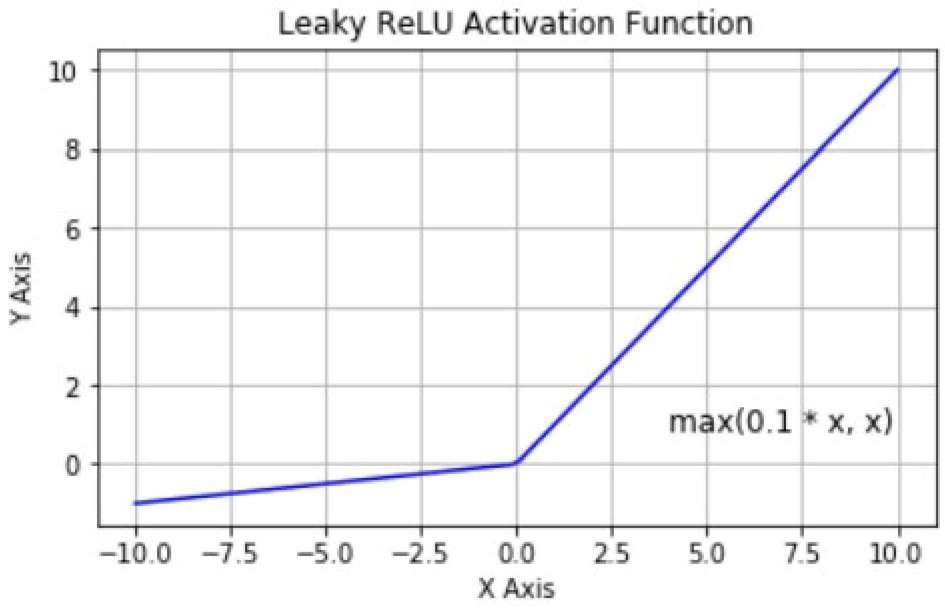


Figure 3: Leaky ReLU Activation Function [26]

2.3 LOSS FUNCTION

The accuracy of the predicted outcome prediction is shown by the loss function. This is performed by comparing the neural network's predicted and observed outputs using different comparison functions. There are several loss functions, just as there are numerous activation functions, and the choice is made dependent on the type of issue the neural network is attempting to solve.

2.3.1 MSE

The **Mean Square Error (MSE)** is a metric that tells us how far apart our predicted values are from our observed values in a regression analysis. A higher MSE shows that the data points are widely spread around the central moment (mean), whereas a lower MSE indicates the reverse. A smaller MSE is preferable since it implies that your data points are evenly distributed around the centre moment (mean). It shows your data's concentrated distribution, the fact that it is not skewed, and, most crucially, that it has fewer errors (measured by the dispersion of the data points from its mean).

The smaller the MSE, the smaller the error, and the better the estimator.

The Mean Squared Error is calculated as:

$$MSE = \frac{1}{n} \sum (actual - forcast)^2 \qquad (9)$$

Where, $\sum$ – a symbol that means "sum", n – sample size, actual – the actual data value, forecast – the predicted data value.

The activation function utilised in this model is the Mean Squared Error (MSE), which is a regularly use as loss function for regression problems, due to the nature of the goal of forecasting the spread of COVID-19. The sum of the squared distance between the predicted and expected values is calculated using this loss function. Outliers have a significant impact on this loss function since the mistakes are squared.

2.4 OPTIMIZATION ALGORITHMS

Deep learning is a machine learning area that is used to tackle complicated tasks like speech recognition and text categorization. An activation function, input, output, hidden layers, loss function, and other components make up a deep learning model. Any deep learning model employs an algorithm to attempt to generalise the data and generate predictions based on previously unknown data. We'll need an algorithm that translates input examples to output examples, as well as an optimization method. When translating inputs to outputs, an optimization method determines the value of the parameters (weights) that minimise the error. These optimization approaches or optimizers have a significant impact on the accuracy of the deep learning model.

We must alter each epoch's weights and minimise the loss function while training the deep learning model. An optimizer is a function or algorithm that alters the characteristics of a neural network, such as its weights and learning rate. As a result, it aids in the reduction of total loss and the improvement of accuracy. Because a deep learning model often has millions of parameters, finding the proper weights for the model is a difficult issue. It necessitates the selection of an appropriate optimization algorithm for your application. As a result, prior to diving further into the topic, you must first comprehend these algorithms. To adjust your weights and learning rate, you can utilise several optimizers. Choosing the best optimizer, on the other hand, is dependent on the application.

2.4.1 Adam Optimizer

Adaptive moment estimate is the source of the word adam. To update network weights during training, this optimization approach is a further development of stochastic gradient descent. Unlike SGD, Adam optimizer modifies the learning rate for each network weight independently, rather than keeping a single learning rate entire training. The Adam optimization algorithm's authors are aware of the advantages of the AdaGrad and RMSProp algorithms, which are both stochastic gradient descent extensions. As a result, the Adam optimizers inherit both Adagrad and RMS prop algorithm characteristics. Instead of using the first moment (mean) like in RMS Prop, Adam employs the second moment of the gradients to modify learning rates. We take the second instant of the gradients to imply the uncentered variance (we don't remove the mean).

3 Proposed Model

3.1 Graph Construction

Both dynamic and static data are included in the input data. Location (e.g., states, counties, etc.), timestamp (e.g., days/weeks), and dynamic characteristics at each location (e.g., the number of current COVID-19 cases and the numbers of other associated ICD codes) make up dynamic data. Static data is a 2D matrix that comprises each place's position as well as its static properties. We also create an attributed graph to represent the spatiotemporal dynamics of the epidemic/pandemic. We model geographic closeness and demographic similarity between distinct sites as edges in a location graph in particular.

To represent the input data, we create an attributed graph G (V; E). A place is represented as a graph node with a feature matrix including both static and dynamic features for all timestamps for that location. The county-level graph has 193 nodes (1 for each county) while if a country has 45 states its state-level graph has 45 nodes (1 for each state).

The edges are built using geographical proximity and node population numbers (i.e., locations). In specifically, we designate the weight of an edge between nodes i and j as $w_{ij} \propto p_i^{\alpha}\ p_j^{\beta}\ exp\ (-\frac{d_{ij}}{r})$, where p_i and p_j are the population sizes of the nodes,

d_{ij} is the geographical distance between the nodes and α, β and r are hyperparameters. The model is based on the assumption that disease transmission patterns are heavily influenced by population movement. If a pair of nodes has a high mobility rate, we anticipate the nodes to have comparable disease spread characteristics. As a result, between such a pair of nodes, our algorithm includes an edge with a weight. It's worth noting that the distance parameter d_{ij} may be used to represent any kind of distance, including the inverse of the volume of air and automobile trips.

Node features (static): Each node n_i includes a four-dimensional static feature vector including static information such as latitude, longitude, population size, and density.

Node features (dynamic): Each node n_i has a set of dynamic characteristics in the form of a matrix. The number of new cases, total cases, current number of hospitalizations, and ICU stays owing to COVID-19, which are derived by aggregating the corresponding procedure codes, are among the dynamic characteristics.

3.2 Graph attention networks are used to model spatio-temporal patterns

To extract spatio-temporal similarity characteristics, we use the GAT model. The main principle behind GAT [18] is to update each node's embedding by aggregating its nearby nodes. To simulate spatio-temporal disease transmission patterns, each place will acquire information from its nearby locations based on mobility. This concern is based on the reality that nearby sites may have varied effects on the infectious state of the focal area. For example, if one city has a huge population and is seeing an increase in infected cases, it may have a significant influence on its neighbouring counties.

We extract spatio-temporal properties from the attributed graph using a two-layer GAT. To create the graph, we use the most recent values from historical data inside a sliding frame. The input characteristics to n_i at time step t are X_t^i, where $X_t^i \in \mathbb{R}^{L_I(F_D+F_S)}$, L_I specifies the length of the input window. Intuitively, we concatenate historical features (ie, L_I days of features) as input at the t-th timestep. The model will employ more past data and trends the longer the L_I is. The graph attention mechanism is then used to construct the node representation $z_t^i \in \mathbb{R}^{F_z}$ for each node, where F_z is the GAT layer's output dimension.

To calculate the z_t^i, we apply the multihead mechanism to compute K independent attention scores using the self-attention technique. A multihead attention mechanism can help the model make more accurate predictions by producing various attention weights. Various attention heads may intuitively focus on different characteristics in the graph to model the locations more completely. The attention weight of the k-th head between 2 nodes i and j is:

$$e_{ij}^{k} = \sigma(W_a^{k}(W_z^{k} X_t^{i} | W_z^{k} X_t^{j})) \qquad (10)$$

where $W_z^k \in \mathbb{R}^{F_z \times |X_t^i|}$ represents the linear transformation weight matrix for the k-th head, which will convert the input dimension to the output dimension. $W_a^k \in \mathbb{R}^{1\times 2F_z}$ represents the attention computation matrix for the k-th head, and $(\cdot \mid \cdot)$ denotes the vector concatenation. r is the nonlinear activation function, and here we use the leaky rectified linear function (LeakyReLU):

$$\sigma(x) = \begin{cases} x, & if\ x \geq 0 \\ 0.01x, & otherwise \end{cases} \qquad (11)$$

It is the same as the GAT [18] model from the beginning. The attention score is then calculated using the softmax function:

$$a_{ij}^{k} = \text{softmax}\left(e_{ij}^{k}\right) = \frac{\exp\left(e_{ij}^{k}\right)}{\sum_{n=1}^{N} \exp\left(e_{in}^{k}\right)} \qquad (12)$$

Each edge of node i will be assigned an attention score, which determines how much information from nearby node j should be pooled. Finally, we add together all of the embedding vectors from different heads to get the final representation z_t^i for node i.

$$z_t^i = \frac{1}{K} \sum_{k=1}^{K} \sum_{j=1}^{N} a_{ij}^{k} \mathbf{W}_z^{k} \mathbf{X}_t^{i} \qquad (13)$$

Where, N stands for the total number of places.

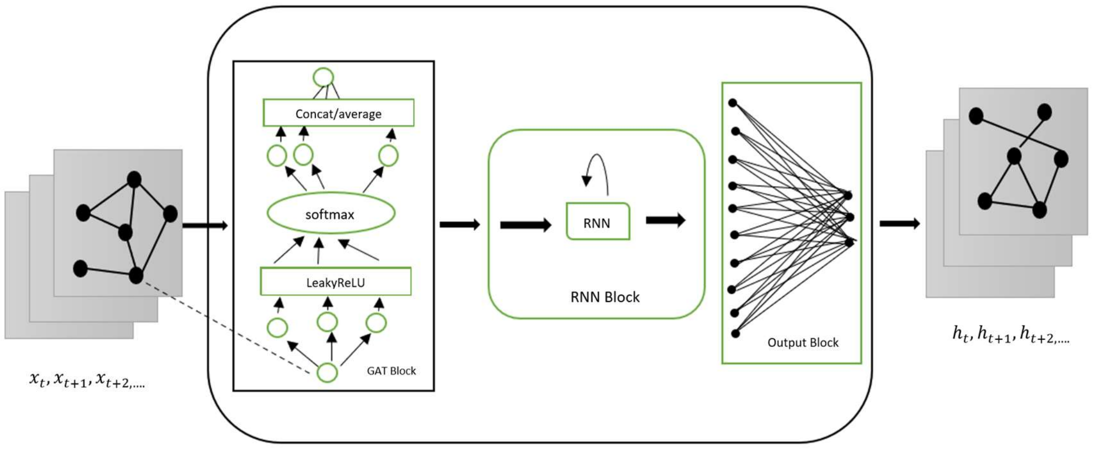


Figure 4: GAT-RNN model architecture

3.3 RECURRENT NEURAL NETWORK(RNN) ARE USED TO MODEL TEMPORAL PATTERN

Pandemic prediction is a challenge that is both spatially and temporally connected. The data on the graph will vary over time. To better forecast future trends, we wish to model the spatio-temporal patterns. To construct embedding for the complete network, first apply the MaxPooling operation, which selects the largest value in each column of a matrix to lower the dimension.

$$\tilde{z}_t = \mathrm{MaxPool}([z_t^0, z_t^1, \ldots, z_t^N]) \qquad (14)$$

where $z_t^0, z_t^1, \ldots\ldots, z_t^N$ is a matrix and the i-th column is z_t^i thus, $\tilde{z}_t$ incorporates the most important features of all nodes extracted from the graph.

For some regions, the pandemic may have just begun, while for others, the epidemic may have reached its height on the same day. As a result, can't use the same model parameters to simulate temporal patterns for all places at the same time. For each location, we create a unique model. Although these sites have the same model structure and graph structure, the model parameters are different. The hybrid gated attention network-RNN model is an end-to-end model, which implies that each location's model will extract the most relevant spatio-temporal patterns from the associated graph in an adaptive manner. we omit location index i to reduce clutter in the following equations, which are all for one unique site.

To learn temporal properties, we feed the graph embedding into an RNN network. A recurrent neural network (RNN) is a sort of neural network that can efficiently simulate temporal sequences and is frequently utilised in numerous sequence analysis applications [6]. The hidden state of the RNN is computed as:

$$h_t = f(h_{t-1}, z_t) \qquad (15)$$

For a given location, the acquired hidden state of the RNN h_t at the t-th timestep comprises both spatial and temporal patterns learnt from real-world data.

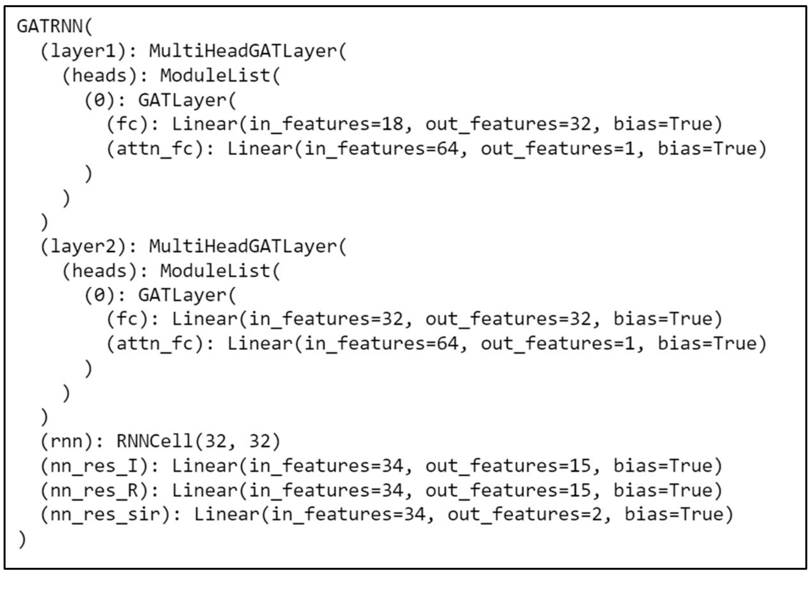
4 DATASET

We used a US county-level dataset in this work that includes COVID-19-related data from two sources: the Johns Hopkins University (JHU) Coronavirus Resource Center [28] and IQVIA's claims data [27]. From March 22, 2020 to December 1, 2020, data from the JHU Coronavirus Resource Center was gathered. It contains information on the number of new cases, active cases, and deaths linked to COVID-19 in various US states. To assure data source accuracy, we chose states with more than 1000 verified cases and the dataset now includes 45 states. We set the number of instances before their respective first record dates to zero for those counties. The claims data from IQVIA comes from the IQVIA US9 Database. From March 22, 2020 to December 1, 2022, we export patient claims and prescription data, from which we calculate the number of hospital and ICU visits, as well as the term-frequency of each medical code by county each day. Supplementary Material contains detailed dataset descriptions. The Supplementary Material has a list of the 48 unique ICD-10 codes associated with COVID-19.

5 MODEL ARCHITECTURE AND TRAINING

5.1 MODEL ARCHITECTURE

To extract spatio-temporal properties from the attributed graph, we employ a two-layer GAT. Then use the RNNCell from Pytorch to predict the new cases for particular timespan of a state. The following diagrams show the architecture of our Graph Attention network–Recurrent Neural network (GAT-RNN) model.



```
GATRNN(
  (layer1): MultiHeadGATLayer(
    (heads): ModuleList(
      (0): GATLayer(
        (fc): Linear(in_features=18, out_features=32, bias=True)
        (attn_fc): Linear(in_features=64, out_features=1, bias=True)
      )
    )
  )
  (layer2): MultiHeadGATLayer(
    (heads): ModuleList(
      (0): GATLayer(
        (fc): Linear(in_features=32, out_features=32, bias=True)
        (attn_fc): Linear(in_features=64, out_features=1, bias=True)
      )
    )
  )
  (rnn): RNNCell(32, 32)
  (nn_res_I): Linear(in_features=34, out_features=15, bias=True)
  (nn_res_R): Linear(in_features=34, out_features=15, bias=True)
  (nn_res_sir): Linear(in_features=34, out_features=2, bias=True)
)
```

Figure 5: GAT-RNN model architecture

5.2 TRAINING AND TESTING

On both a county and state level, we forecast the number of new cases in the future. We set the prediction window L_P 15 to test GAT-RNN's ability to make both long-term and short-term predictions (i.e., predict for future 15 days). All training sets will begin on March 22nd, and all test sets will begin on November 15th. To select model hyperparameters, we split L_P days from the training sets into evaluation sets. All locations are utilized in the training and testing set, which is separated along the time dimension, with early time windows used for training and later time windows used for testing. All of the models are trained with the same data and reviewed and tested using the same prediction time frame.

5.3 HYPERPARAMETER

The more sophisticated architecture of the GAT-RNN model necessitated the usage of more hyperparameters. The activation function, optimizer, Epoch, Loss function are the hyperparameters.

Hyperparameter	**Value**
GAT activation function	LeakyReLU
RNN activation function	tanH
Optimizer	Adam
Epoch	100
Loss Function	MSE

Table 1: Hyperparameters for GAT-RNN model

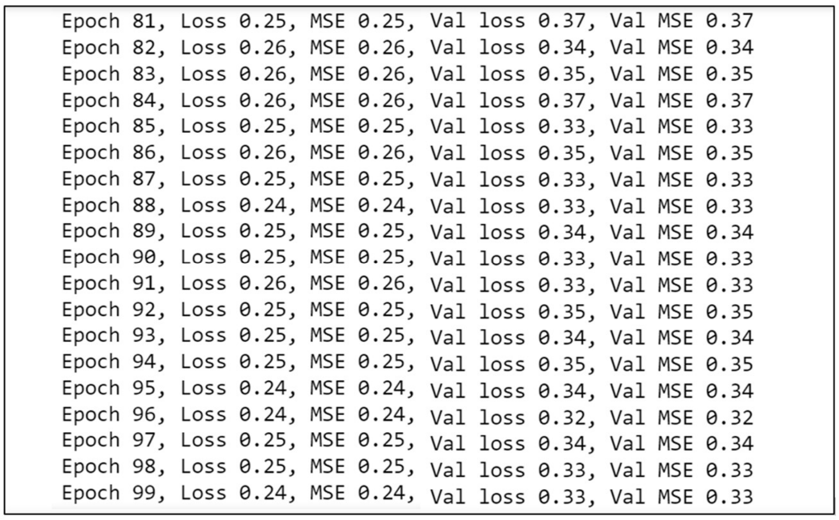
6 EXPERIMENTAL RESULTS

6.1 RUN TIME

Another important aspect of the training process is the runtime. The model used to predict the spread of COVID-19 have can affect the amount of time taken to train the model. Although the runtime of the training process is very dependent on hardware and processor speeds. Here in our machine, we use 11th Gen Intel(R) Core(TM) i5-11300H @ 3.10GHz 3.11 GHz as processor, 16.0 GB (15.8 GB usable) of RAM and 64-bit operating system Windows 11, x64-based processor. The training times from this machine approx. 90 sec. Runtimes are important to evaluate because they ultimately determine how quickly a model can return case count predictions and how much time communities have to prepare during a pandemic. This is an important factor to acknowledge when comparing the performances of other models.

6.2 MODEL PERFORMANCE

We utilize the MSE as our loss function, as previously mentioned. That can forecast the number of new cases of a particular US state using US COVID-19 data. In this model predicts disease transmission on test data from November 15th, 2020 to December 1st, 2020, using COVID-19 case counts from March 22nd, 2020 to November 14th, 2020 as training data. As previously indicated, we used the Adam optimizer in our model. We calculate the model's training and testing accuracy when training the model on COVID-19 data. The model estimated the loss over train and test data for 100 epochs. It readjuste the model after each computation to optimize the loss, and so on.



```
Epoch 81, Loss 0.25, MSE 0.25, Val loss 0.37, Val MSE 0.37
Epoch 82, Loss 0.26, MSE 0.26, Val loss 0.34, Val MSE 0.34
Epoch 83, Loss 0.26, MSE 0.26, Val loss 0.35, Val MSE 0.35
Epoch 84, Loss 0.26, MSE 0.26, Val loss 0.37, Val MSE 0.37
Epoch 85, Loss 0.25, MSE 0.25, Val loss 0.33, Val MSE 0.33
Epoch 86, Loss 0.26, MSE 0.26, Val loss 0.35, Val MSE 0.35
Epoch 87, Loss 0.25, MSE 0.25, Val loss 0.33, Val MSE 0.33
Epoch 88, Loss 0.24, MSE 0.24, Val loss 0.33, Val MSE 0.33
Epoch 89, Loss 0.25, MSE 0.25, Val loss 0.34, Val MSE 0.34
Epoch 90, Loss 0.25, MSE 0.25, Val loss 0.33, Val MSE 0.33
Epoch 91, Loss 0.26, MSE 0.26, Val loss 0.33, Val MSE 0.33
Epoch 92, Loss 0.25, MSE 0.25, Val loss 0.35, Val MSE 0.35
Epoch 93, Loss 0.25, MSE 0.25, Val loss 0.34, Val MSE 0.34
Epoch 94, Loss 0.25, MSE 0.25, Val loss 0.35, Val MSE 0.35
Epoch 95, Loss 0.24, MSE 0.24, Val loss 0.34, Val MSE 0.34
Epoch 96, Loss 0.24, MSE 0.24, Val loss 0.32, Val MSE 0.32
Epoch 97, Loss 0.25, MSE 0.25, Val loss 0.34, Val MSE 0.34
Epoch 98, Loss 0.25, MSE 0.25, Val loss 0.33, Val MSE 0.33
Epoch 99, Loss 0.24, MSE 0.24, Val loss 0.33, Val MSE 0.33
```

Figure 6: training and testing loss

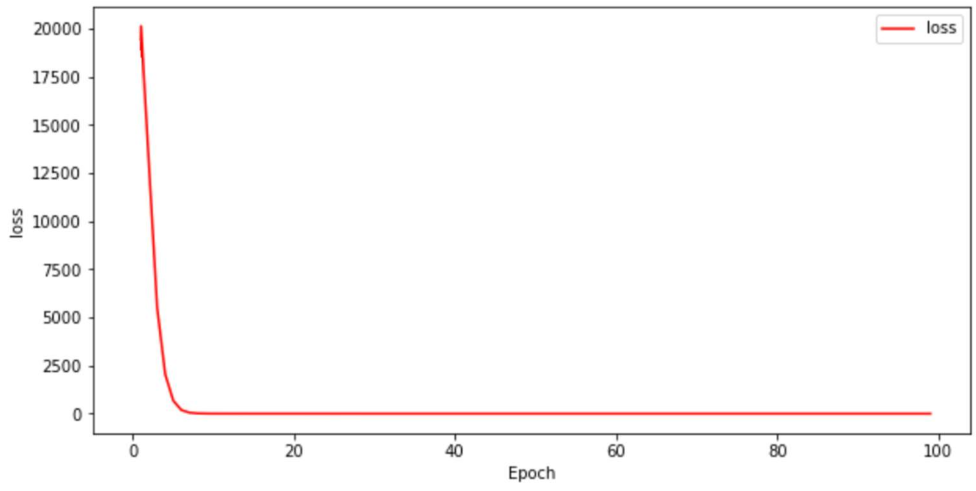


Figure 7: GAT-RNN model training loss in 100 epochs

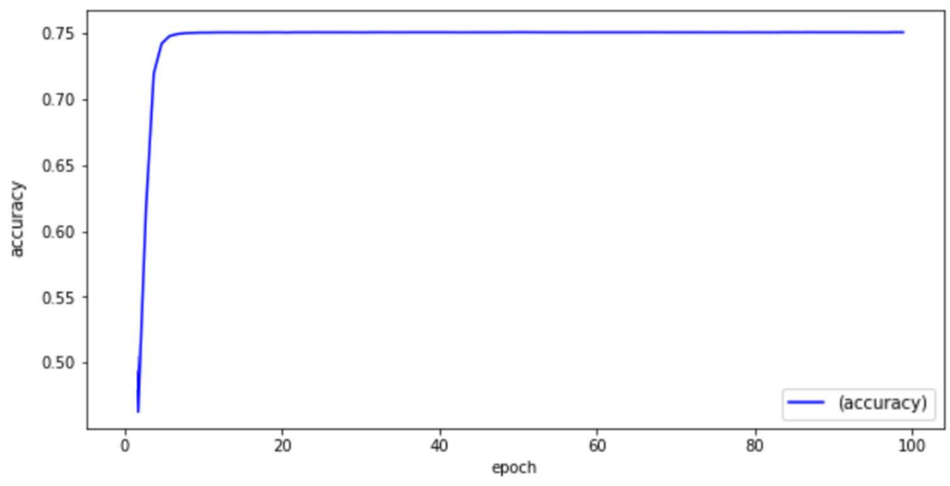


Figure 8: GAT-RNN model accuracy in 100 epochs

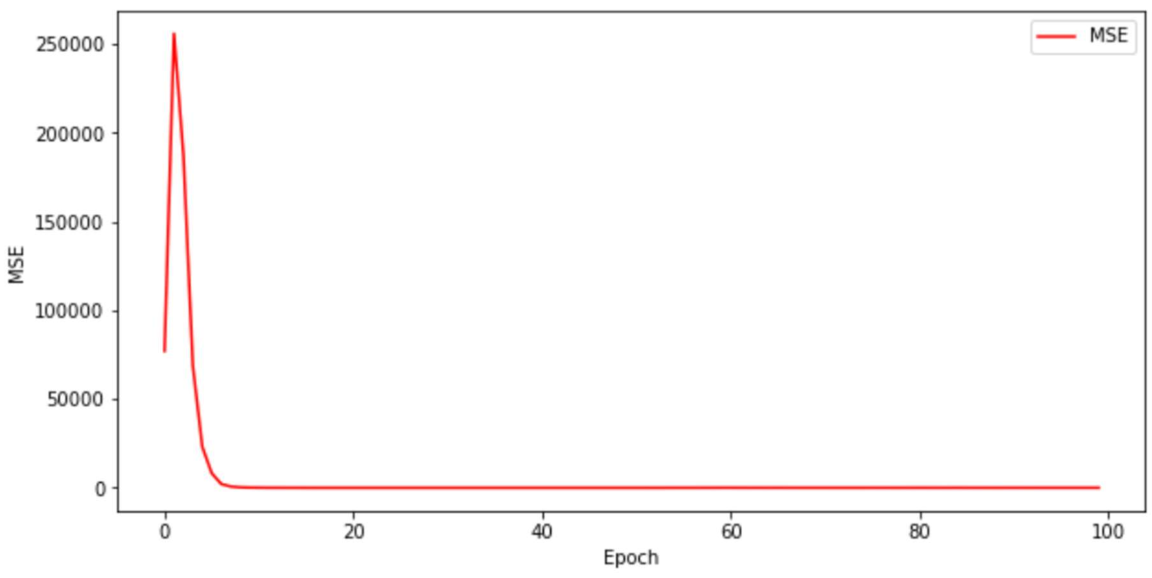


Figure 9: MSE in 100 epochs

6.3 Evaluation Metrics

Here, the errors are calculated based on New York state test data from date 15/11/20 to 1/12/20.

The Mean Squared Error (MSE) is perhaps the simplest and most common loss function, often taught in introductory Machine Learning courses. In our model MSE error is:

MSE = **499020**

The Root Mean Square Error (RMSE) (also known as the root mean square deviation, RMSD) is a commonly used measure of the difference between values predicted by a model and actual values observed in the modelled environment.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}} \quad (16)$$

The Root Mean Square Error (RMSE) is a measurement of the difference in error between two data sets. In our model RMSE error is:

RMSE = $\sqrt{MSE}$ = $\sqrt{499020}$ = 7**06.41**

The mean squared logarithmic error (MSLE) is a measurement of the ratio between actual and predicted values. It's a kind of MSE that's just interested in the percentual difference. In our model MSLE error is:

MSLE = **0.061723713** = **0.062**

Root-mean-square logarithmic error (RMSLE) is the root mean squared error of the log-transformed predicted and log-transformed actual values. RMSLE is a metric that measures the ratio of what is predicted and what really happened. In our model MSLE error is:

RMSLE = $\sqrt{MSLE}$ = $\sqrt{0.062}$ = **0.248997992 = 0.25**

Mean absolute percentage error (MAPE) is a measure of prediction accuracy of a forecasting method. It usually expresses the accuracy as a ratio defined by the formula below:

$$\text{MAPE} = \frac{100}{n} \sum_{t=1}^{n} \left| \frac{A_t - F_t}{A_t} \right| \quad (17)$$

A_t is the actual value and F_t is the forecast value. The absolute value in this ratio is summed for every forecasted point in time and divided by the number of fitted points n. In our model MAPE error is:

MAPE = **0.21435834 = 0.21**

6.4 RESULTS

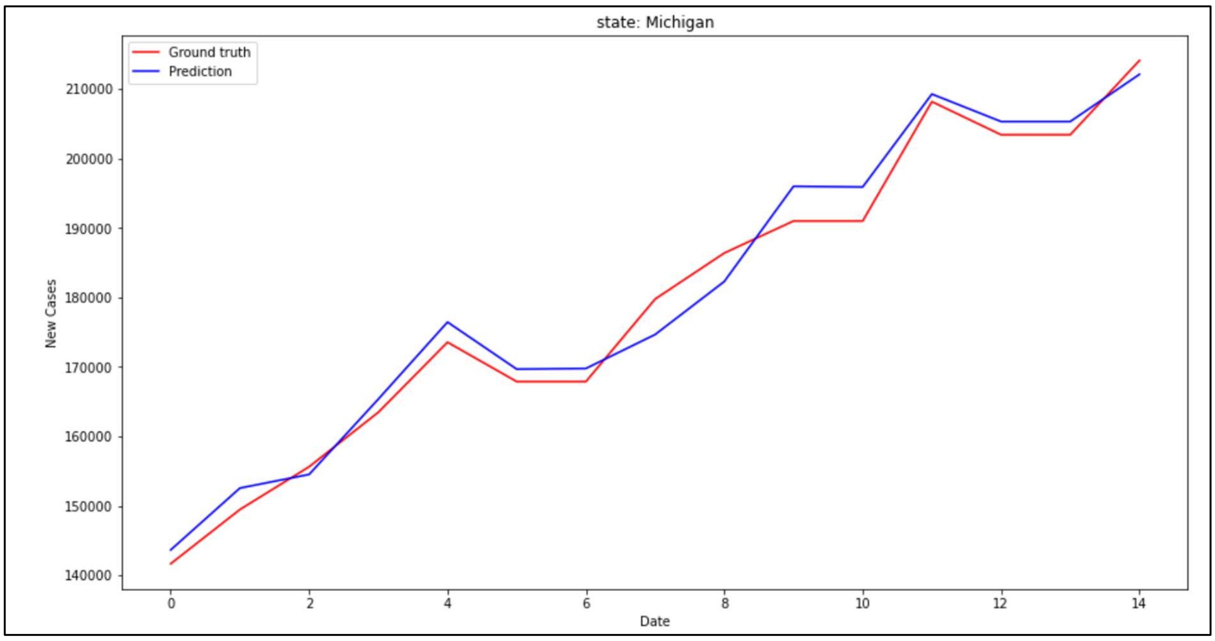


Figure 10: Michigan state New Cases prediction from 15/11/20 to 1/12/20

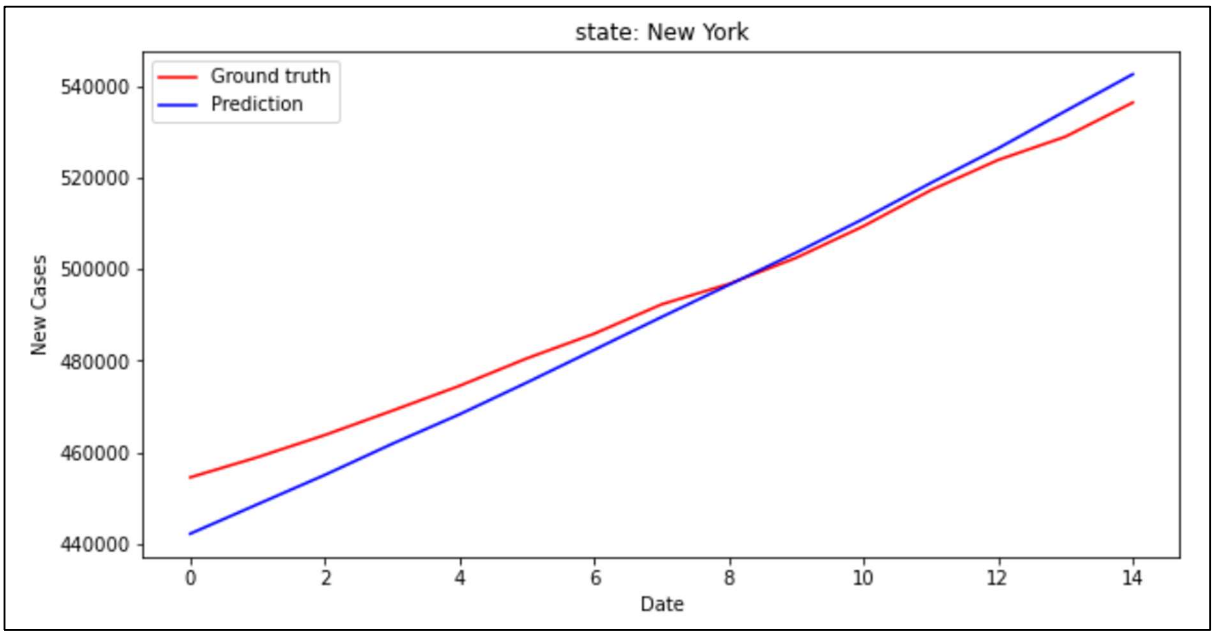


Figure 11: New York state New Cases prediction from 15/11/20 to 1/12/20

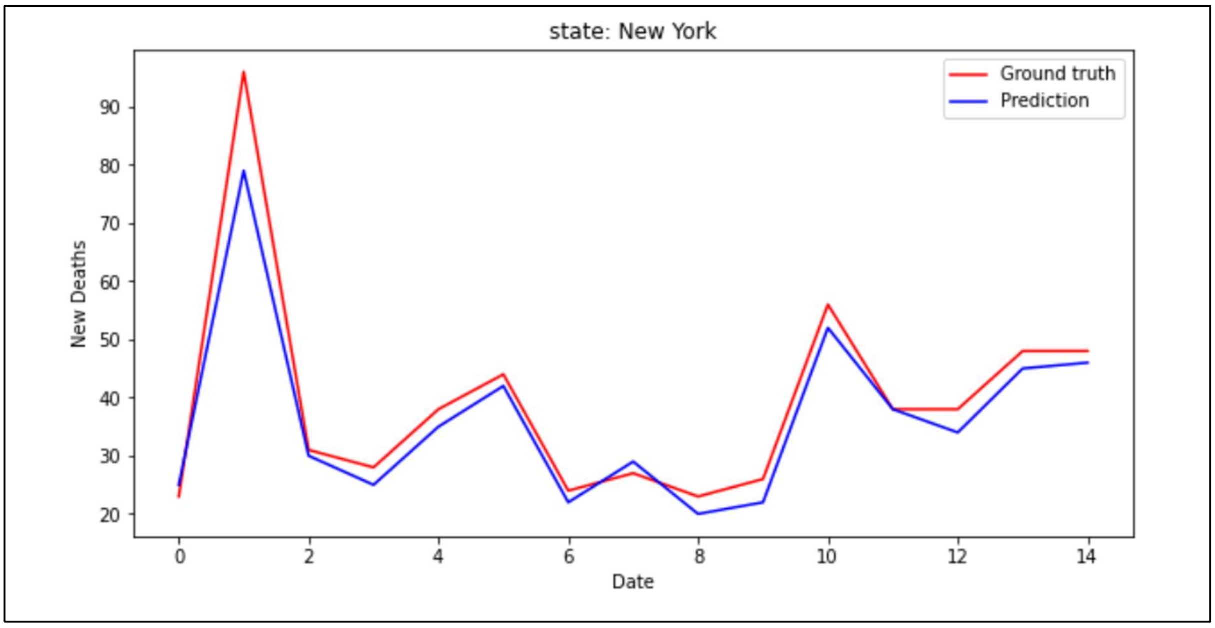


Figure 12: New York state new death prediction from 15/11/20 to 1/12/20

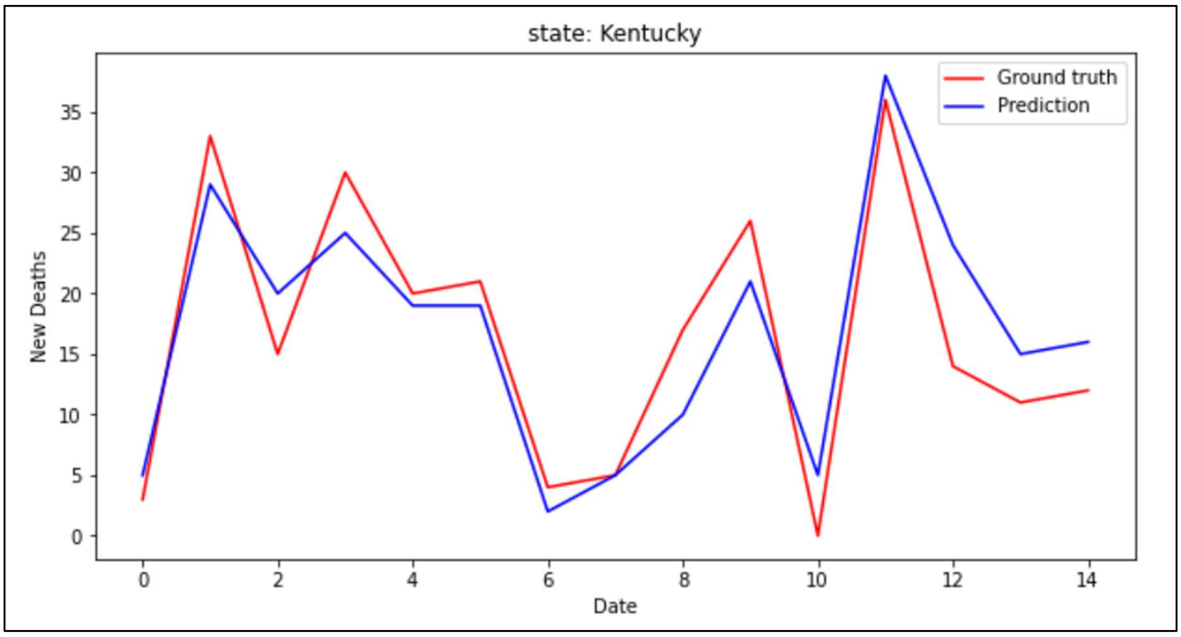


Figure 13: Kentucky state new death prediction from 15/11/20 to 1/12/20

6.5 DISCUSSION

We draw the predicted curve of new cases for 2 states, New York and Michigan. the performance improvement is much greater, GAT-RNN can achieve at lower MSE compared to the traditional models as shown in Figures 10 and Figures 11, the curve also fits the actual trend better for states. One obvious drawback of traditional models is the overfitting issue because these models do not incorporate the influence and interdependency of transmission between geographic regions. The features of infectious disease transmission in one location are unlikely to be dissociated from those in nearby areas unless there are obstacles to contact between the regions such as geography (rivers with few bridges or mountain ranges with restricted road connections) or regulated boundaries. In the United States, such decoupling between counties is uncommon.

The proposed model is combines a time-series forecasting model and a graph neural network model which is very uncommon machine learning algorithms. The GAT-RNN model is an extension of the baseline SRI, SEIR and was expected to perform better due to the model's integration of the geography of the world in making its predictions. The GAT-RNN's sophisticated architecture had an impact on the result we obtained. Looking at the graph data structure of the states over the US, there are a total of 165 edges connecting the 52 states, which certainly played a role in its long training time. It is a very complicated network to learn. It performs comparatively well in predicting the transmission of COVID-19 on a macro-scale. Our research reveals several insights about the pandemic.

Whereas other studies mentioned that used Graph Neural Networks in their macro-scale studies are able to incorporate mobility data into their neural networks, mobility data is not available in a format that could be used by our GAT-RNN model. The integration of mobility data among the towns could possibly improve the results of this model.

The lack of mobility information in the Graph Attention Network may also have prevented the GAT-RNN model from being exploited to its full potential. Given its propensity to include spatial characteristics into its learning, it may not have been the ideal option for this specific macro-scale time-series forecasting issue, but it is a very robust model that is definitely worth investigating.

7 CONCLUSION

7.1 LIMITATION

Although deep learning-based approaches beat standard statistical methods in a variety of time series analysis and prediction applications, our model has two main limitations. The prediction window setting in our technique is the first constraint. Traditional methods fit the model and construct the complete curve using all prior training data to reduce the impact of data volatility. In training time, however, our model splits past data into prediction windows. GAT-RNN can now dynamically predict pandemic progressions thanks to this configuration. It is difficult for the GAT-RNN model to learn correct and consistent transmission and recovery rates if the number of cases changes dramatically owing to inaccuracies in the data gathering method. This problem can be further remedied by smoothing such aberrant data points with dynamic data smoothing.

The second limitation is about data quality for constructing the attribute graph. Indeed, because to report delays or other factors, ICD codes for nodal traits may not always represent true pandemic status. Incorporating other data sources, such as traffic information or mobile geolocation monitoring, can also improve the edge mobility computation. Furthermore, the dynamic and static properties are processed jointly without taking into account their differences. Future research might include more data sources and expand the transmission dynamics limitations in GAT-RNN to improve prediction performance with richer data and handle varied data types more efficiently. By adding relevant statistics such as COVID-19 related therapy codes and ICU codes to the data, our model may readily be used for mortality or hospitalization prediction tasks.

7.2 CONCLUSION AND FUTURE WORKS

In this section, we will discuss the advantages and also the limitations of our model. We propose a spatio-temporal hybrid gated attention network RNN model (GAT-RNN) for predicting the COVID-19 pandemic in this thesis. We connect locations (for example, a county or a state) to nodes on a graph. We generate nodes using a collection of static and dynamic features taken from a variety of real-world evidence, including real-world medical claims data, and edges using geographical latitude longitude and demographic similarity across sites. We employ a GAT to account for the varied effect of a node's various nearby sites and forecast the number of infected patients for a certain time period in the future. Transmission dynamics restrictions are also imposed on predictions based on transmission dynamics. GAT-RNN beats both the classic SIR and SEIR models, as well as other deep learning approaches, in terms of prediction performance, and has less overfitting difficulties in the early stages of the epidemic. We anticipate that our model will aid governments and researchers in better allocating medical resources and developing strategies to combat the epidemic sooner.

As future study, our approach may simply be expanded to predict COVID-19 hospitalization. We plan to integrate additional factors such as age group, outside traveller into the prediction to further improve the performance. Additionally, new technologies to enhance the capability of finding missing raw data will be incorporated.

8 REFERENCES

1. Wikipedia. COVID-19 pandemic data. Secondary COVID-19 pandemic data 2020. https://en.wikipedia.org/wiki/Template: COVID-19_pandemic_data Accessed July 2, 2020.

2. Pei S, Shaman J. Initial Simulation of SARS-CoV2 Spread and Intervention Effects in the Continental US. *medRxiv* 2020. 03.21.20040303; doi: 10.1101/2020.03.21.20040303.

3. Yang Z, Zeng Z, Wang K, et al. Modified SEIR and AI prediction of the epidemics trend of COVID-19 in China under public health interventions. *J Thorac Dis* 2020; 12 (3): 165–74.

4. B. Xu, N. Wang, T. Chen, and M. Li, ``Empirical evaluation of rectified activations in convolutional network," 2015, *arXiv:1505.00853*.

5. T. N. Kipf and M. Welling, ``Semi-supervised classifcation with graph convolutional networks," 2016, *arXiv:1609.02907*. [Online]. Available: https://arxiv.org/abs/1609.02907

6. Tokgo¨z A, Unal GA. RNN based time series approach for forecasting € turkish electricity load. In: 2018 *26th Signal Processing and Communications Applications Conference (SIU)*. Izmir, Turkey: IEEE.

7. United Nations. 2020. COVID-19 to slash global economic output by 8.5 trillion over next two years. https://www.un.org/development/desa/en/news/policy/wesp-mid-2020-report.html (visited on 6/4/2020). (2020).

8. Roberts M, Andreasen V, Lloyd A, Pellis L. Nine challenges for deterministic

epidemic models. Epidemics 2015; 10: 49–53. 5. Durbin J, Koopman SJ. Time Series Analysis by State Space Methods. Oxford, UK: Oxford University Press; 2012.

9. F. Rosenblatt, “The perceptron: A probabilistic model for information storage and organization in the brain.,” *Psychol. Rev.*, vol. 65, no. 6, pp. 386–408, 1958, doi: 10.1037/h0042519.

10. S. K. Pal and S. Mitra, “Multilayer perceptron, fuzzy sets, and classification,” *IEEE Trans. Neural Netw.*, vol. 3, no. 5, pp. 683–697, Sep. 1992, doi: 10.1109/72.159058.

11. S. Kombrink, T. Mikolov, M. Karafiat, and L. Burget, “Recurrent Neural Network Based Language Modeling in Meeting Recognition,” p. 4.

12. J. L. ELMAN, “Finding structure in time,” 1990.

13. F. Scarselli, M. Gori, Ah Chung Tsoi, M. Hagenbuchner, and G. Monfardini, “The Graph Neural Network Model,” *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61– 80, Jan. 2009, doi: 10.1109/TNN.2008.2005605.

14. Durbin J, Koopman SJ. *Time Series Analysis by State Space Methods*. Oxford, UK: Oxford University Press; 2012.

15. Wang L, Chen J, Marathe M. DEFSI: deep learning based epidemic forecasting with synthetic information. In: *Proceedings of the AAAI Conference on Artificial Intelligence*; February 1, 2019; Honolulu, Hawaii.

16. Deng S, Wang S, Rangwala H, Wang L, Ning Y. Graph message passing with cross-location attentions for long-term ILI prediction. *arXiv preprint arXiv*: 1912.10202; 2019.

17. Kapoor A, Ben X, Liu L, et al. Examining COVID-19 Forecasting using Spatio-Temporal Graph Neural Networks. *arXiv preprint arXiv*: 2007.03113; 2020.

18. Velickovic P, Cucurull G, Casanova A, Romero A, Lio P, Bengio Y. Graph Attention Networks. *International Conference on Learning Representations*; April 30, 2018; Vancouver, Canada.

19. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, ``Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998_6008.

20. CJ Murray et al. 2020. Forecasting COVID-19 impact on hospital bed-days, ICU-days, ventilator-days and deaths by US state in the next 4 months. (2020).

21. Nuria Oliver, Bruno Lepri, Harald Sterly, Renaud Lambiotte, Sébastien Deletaille, Marco De Nadai, Emmanuel Letouzé, Albert Ali Salah, Richard Benjamins, Ciro Cattuto, et al. 2020. Mobile phone data for informing public health actions across the COVID-19 pandemic life cycle. (2020).

22. Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. 2018. Relational inductive biases, deep learning and graph networks. arXiv preprint arXiv:1806.01261 (2018).

23. Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. 2019. A comprehensive survey on graph neural networks. arXiv preprint arXiv:1901.00596 (2019).

24. RNN Cell https://theaisummer.com/understanding-lstm/

25. TanH activation function https://www.baeldung.com/cs/sigmoid-vs-tanh-functions

26. leakyReLU https://www.researchgate.net/figure/Plot-of-the-LeakyReLU-function_fig9_325226633

27. IQVIA. Real World Data and Insights. Secondary Real World Data and Insights 2020. https://www.iqvia.com/solutions/real-world-evidence/realworld-data-and-insights Accessed July 2, 2020.

28. https://github.com/CSSEGISandData/COVID-19