Data-Driven Time Series Forecasting for Social Studies Using Spatio-Temporal Graph Neural Networks

Yi-Fan Li
yli@utdallas.edu
Erik Jonsson School of Engineering and Computer Science
University of Texas at Dallas
Richardson, TX

Bo Dong
bo.dong3@utdallas.edu
Erik Jonsson School of Engineering and Computer Science
University of Texas at Dallas
Richardson, TX

Latifur Khan
lkhan@utdallas.edu
Erik Jonsson School of Engineering and Computer Science
University of Texas at Dallas
Richardson, TX

Bhavani Thuraisingham
bhavani.thuraisingham@utdallas.edu
Erik Jonsson School of Engineering and Computer Science
University of Texas at Dallas
Richardson, TX

Patrick T. Brandt
pbrandt@utdallas.edu
School of Economic, Political, and Policy Sciences
University of Texas at Dallas
Richardson, TX

Vito J. D'Orazio
dorazio@utdallas.edu
School of Economic, Political, and Policy Sciences
University of Texas at Dallas
Richardson, TX

ABSTRACT

Time series forecasting with additional spatial information has attracted a tremendous amount of attention in recent research, due to its importance in various real-world applications on social studies, such as conflict prediction and pandemic forecasting. Conventional machine learning methods either consider temporal dependencies only, or treat spatial and temporal relations as two separate autoregressive models, namely, space-time autoregressive models. Such methods suffer when it comes to long-term forecasting or predictions for large-scale areas, due to the high nonlinearity and complexity of spatio-temporal data. In this paper, we propose to address these challenges using spatio-temporal graph neural networks. Empirical results on Violence Early Warning System (ViEWS) dataset and U.S. Covid-19 dataset indicate that our method significantly improved performance over the baseline approaches.

CCS CONCEPTS

• **Human-centered computing** → **Social engineering (social sciences)**; • **Mathematics of computing** → *Time series analysis*; • **Computing methodologies** → *Neural networks*.

KEYWORDS

Spatio-temporal modeling, Graph neural networks, Conflict prediction, Pandemic forecasting

ACM Reference Format:
Yi-Fan Li, Bo Dong, Latifur Khan, Bhavani Thuraisingham, Patrick T. Brandt, and Vito J. D'Orazio. 2021. Data-Driven Time Series Forecasting for Social Studies Using Spatio-Temporal Graph Neural Networks. In *Conference on Information Technology for Social Good (GoodIT '21), September 9–11, 2021, Roma, Italy.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3462203.3475929

GoodIT '21, September 9–11, 2021, Roma, Italy

ACM ISBN 978-1-4503-8478-0/21/09…$15.00
https://doi.org/10.1145/3462203.3475929

1 INTRODUCTION

Data-driven time series forecasting methods using machine learning techniques have become one of the most important approaches in modern social studies. For example, generating models to predict the state-based conflicts in Africa, including the fatalities associated with these conflicts, are extremely significant topics in political science research, such as Violence Early Warning System (ViEWS) [8]. By getting accurate forecastings, it could provide international communities with early warning signals of the humanitarian crisis, and it becomes possible for international organizations to intervene and prevent the conflict from happening beforehand. In the case of Covid-19 spread, this pandemic has led to unprecedented social consequence and upheaval. Due to the drastic impact of Covid-19 to the whole mankind, a good estimation about infected cases and death cases in a timely manner could be critical for a better spread control, and are helpful for policy and guidance making [1]. In both examples mentioned above, recent advances in technologies enable researchers to obtain a tremendous amount of data. Meanwhile, data-driven models with the underlying machine learning techniques, appear to be more powerful to use these collected data and make predictions even with very limited prior domain knowledge.

Existing data-driven forecasting methods normally follow two lines of sight: statistical methods and machine learning methods. For statistical methods, the autoregressive integrated moving average (ARIMA) model and its variants are one of the most consolidated approaches in this category [20]. However, this approach assumes the time sequences to be stationary along time, and it doesn't consider the additional spatial correlations underlying the data. Recently, widely-used machine learning frameworks, such as random forest regressor, or deep learning approaches (e.g., recurrent neural networks and its variants), have shown their superior performance in making sequence predictions [13]. Compared to classical statistical methods, machine learning models could capture higher order of nonlinearity, due to their more complex model structure [4, 5]. Take

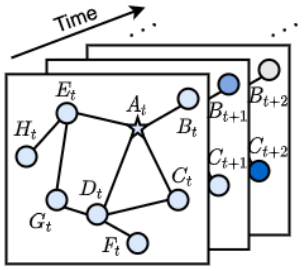

Figure 1: Different nodes (counties) at the same time step share spatial dependencies, while the same node at distinct time steps shares temporal dependencies. Therefore the Covid-19 spread forecasting can be formulated as a spatio-temporal modeling problem.

the recurrent neural networks (RNN) for example, it models the sequence data step by step and allows previous outputs to be used as inputs while having hidden states. The neural network module along with the weights in it is shared between steps [16]. Therefore, RNN model could remember information from previous time steps. But one of the drawbacks of RNN models is that normally they can not quite handle the long-term dependencies within the sequence. On the other hand, its variant, which is called long-short term memory (LSTM) network, tackles RNN's shortcomings and designs a cell state in addition to the hidden state to pass long-term information along the sequences [9]. Still, both RNN and LSTM models consider temporal dependencies only and ignore the spatial dependencies.

In the scenario of real-world social studies, it is worth noticing that normally data could include both spatial and temporal dependencies at the same time. For example, for the Covid-19 spread prediction, for a certain county in the US, there are local spreads of the virus, which can be considered as temporal relations. Meanwhile, since the movement activities in adjacent counties could be relatively frequent, so does the virus. Therefore, there are also spatial relations between nearby counties in the Covid-19 spread case, which should be considered during the modeling process as well. Such spatial-temporal can be demonstrated in Fig. 1. As we mentioned before, conventional time series forecasting models fail to capture the spatial dependencies in the model, and struggles to model the complex nonlinearities of temporal dependencies. We propose to use spatio-temporal graph neural networks to address the aforementioned challenges in this data-driven time-series forecasting problems for social studies. The key idea here is to use a single graph neural network (GNN) structure to model both spatial and temporal dependencies simultaneously. The main contributions of this work are threefold:

- We model the spatial dynamics of adjacent areas in a general graph, instead of treating them separately (e.g. segments.) Meanwhile, we consider use temporal convolution techniques along the time axis to model the temporal dependencies. By combining these two modules altogether in a deep neural network (DNN) block, both spatial and temporal dependencies can be captures at the same time.
- We concatenate the proposed DNN blocks together in a single deep learning architecture, so that the designed framework can model the spatial dynamics from multiple hops away, and capture much longer temporal dependencies from the past. Such architecture is extremely flexible, as it is possible to define the order of both dependencies by simply modifying the number of concatenated modules (layers).
- We compare the proposed framework with other state-of-the-art (SOTA) methods, including temporal-only models and spatio-temporal deep learning models in two real-world datasets. Experiment results demonstrate the superiority of our method compared to basedline methods.

2 METHOD

A typical time series forecasting problem aims to predict future length-Q steps based on previous P observations. In this application we map our training data x_t to predicted values y_t using some forecast function $f(\cdot)$:

$$[X_{(t-P+1)}, \cdots, X_{(t)}] \xrightarrow{f(\cdot)} [y_{(t+1)}, \cdots, y_{(t+Q)}] \tag{1}$$

Given the geographical connections between grid cells in ViEWS, or the counties adjacent relations in U.S, the map can be defined as an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{A})$, where $\mathcal{V}$ represents a set of locations with $|\mathcal{V}| = N$; $\mathcal{E}$ is a set of edges indicating the connectivity between locations, and; $\mathbf{A} \in \mathbb{R}^{N \times N}$ denotes the adjacency matrix of graph $\mathcal{G}$. Notably, the overall raw inputs to our model are denoted by $X \in \mathbb{R}^{P \times N \times d}$ where d is the dimension of features. Here, the solution for the time series forecasting problem requires considering both the spatial and temporal dependencies in the data. In consequence, our approach splits the solution into two sets of convolutional neural networks over space and time, and combines them with a layering technique.

2.1 Spatial Dependency Modeling

The Graph Convolutional Network (GCN) is a model for mining graph structured data, (e.g., social networks, online advertising, etc. [10]). In this implementation, we define the geographic map as an undirected graph, such that the nodes of the graph are the locations, and edges of the graph represent the adjacency relations between locations. Mathematically, such relationship can be referred to as an *adjacency matrix* defined as $\mathbf{A}$. The GCN module handles the spatial dependencies by passing the information to each location from its surrounding locations using a weighted average (convolutions) of their features.

Following Kipf and Welling [10], the spatial dependency model for each time step $t \leq P$ has two inputs in this GCN module. The first one is a variable $X_{sp} \in \mathbb{R}^{N \times d}$, where d is the dimension of features for each location. The second put is the adjacency matrix $\mathbf{A}$ of the graph. In adjacency matrix, locations are sorted by ascending order regarding its ID to keep the consistency during propagation. For example, $A_{ij} = 1$ means that ith cell is adjacent to jth cell, while $A_{ij} = 0$ means that these two locations are not directly connected to each other.

For each time step $t \leq P$, the layer-wise propagation rule of the GCN relationships $H^{(l+1)}$ for layer l of the neural network is :

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right) \tag{2}$$

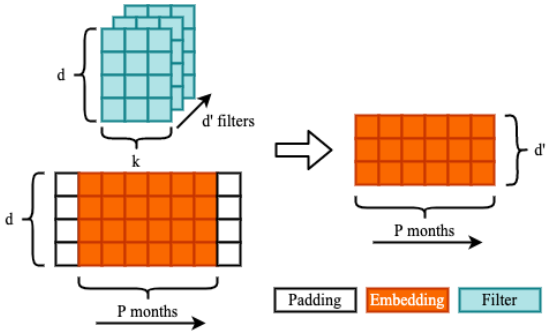

Figure 2: Demonstration of 1D Convolution. By applying tricks such as padding, we keep the time steps of the outputs the same as inputs. The feature dimension of outputs is determined by the number of filters. In this example, $d = 4, d' = 3, P = 6$ and $k = 3$

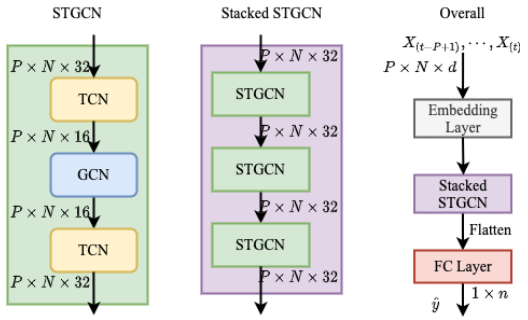

Figure 3: Design of STGCN-TCN module using ViEWS dataset as an example. This framework consists three STGCN blocks with a fully-connected (FC) layer by the end to generate the forecast outputs.

where $H^{(l)} \in \mathbb{R}^{N \times d}$ is the activation matrix in the l^{th} layer with $H^{(0)} = X$. With an identity matrix I_N of N dimension, the $\tilde{A} = A + I_N$ is the adjacency matrix of the undirected graph $\mathcal{G}$ with self-connections. Here $\tilde{D}$ is the degree matrix which can be represented as $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$ and $W^{(l)} \in \mathbb{R}^{d \times d'}$ is a layer-specific trainable weight matrix. Here, d and d' represent the number of feature dimensions at the l^{th} layer for the input and output, respectively. Finally, we choose ReLU as the activation function in the graph convolutions during propagation.

Equation 2 implies a two-step propagation rule for the GCN: (1) aggregate the information from each cell's neighbors; (2) update the neural network. The $\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}$ term in Equation 2 represents the graph convolution, passing the neighbor information to each cell, and then aggregate them based on the connection weights. The aggregated matrix is put into a standard neural network, which multiplies it by the trainable weight matrix $W^{(l)}$, and feeds it to the activation function $\sigma(\cdot)$.

2.2 Temporal Dependency Modeling

For the temporal aspects of the model, Temporal Convolution Networks (TCN) and Recurrent Neural Networks (RNN) are two main deep learning approaches [12]. In this work, we propose to develop temporal models based on TCN, and the temporal inputs for each node are $X_{tp} \in \mathbb{R}^{P \times d}$.

The basic idea of TCN-based models is that temporal information is aggregated by learnable convolution filters–similar to moving averages [22]. Figure 2 shows the basic idea of one dimensional temporal convolution with a kernel of size k. Basically, this operation allows the model to explore the temporal dependency of each node for k steps at a time. The convolutional kernel K_t is designed to map the input X_{tp} to a embedding. In order to keep time steps embeddings consistent along time as P, we pad zero to both beginning and end of the window so it stays the same size.

Thus the dimension of embedding after this convolution step is $P \times d'$. Then, an activation function is applied to this embedding and this whole process can be abstracted as:

$$H^{(l+1)} = \sigma\left(\text{conv}(H^{(l)}, W^{(l)})\right) \tag{3}$$

where conv denotes the convolution operation, $W^{(l)}$ is the trainable weight representing convolutional kernel here with dimension of $k \times d \times d'$.

2.3 Combined Spatio-Temporal Model

We combine the two aforementioned modeling modules together to form our STGCN-TCN framework. Overall, the STGCN-TCN model is composed of a stacked modules, and a Fully Connected (FC) layer at the end to generate predictions. We stack three identical STGCN modules altogether to jointly process graph-structured time series. Note it is possible to stack more STGCN modules depending on the requirement of the task and the input dimension for the first STGCN block is different from latter ones as the input features here are the raw data with d dimensions.

Figure 3 shows, in each STGCN block, how we stack the sub-modules in TCN-GCN-TCN order. The spatial layer is between the two temporal layers and links those two temporal layers together. Considering the dimensions of both temporal and spatial layers, we purposely design it similar to a "sandwich" structure. This trick is also know as "bottleneck strategy", which is very useful when designing a neural network. Having such design encourages the network to compress feature representations to best fit in the available space, to get the best fitting of the loss function during training. Also, such design enforce the ability of model to learn from the data other than remember the data, which could help with the generalization capability of the model. Based on existing parameters, each STGCN module is capable of aggregating 1-hop of spatial dependencies, and three steps of temporal dependencies. Since the stacked STGCN model has three STGCN modules in it sequentially, overall our proposed framework can aggregate up to 3-hops of spatial dependencies, and up to 5 steps of temporal dependencies in such block. These are clearly tuneable parameters in the model.

By generating the hidden representations for each of the node (location) after the combined spatio-temporal module, we feed those embeddings to fully-connected (FC) layer to generate predictions afterwards. For this model, we generate the Q steps of the predictions at once, which makes the dimension of outputs as $N \times Q$.

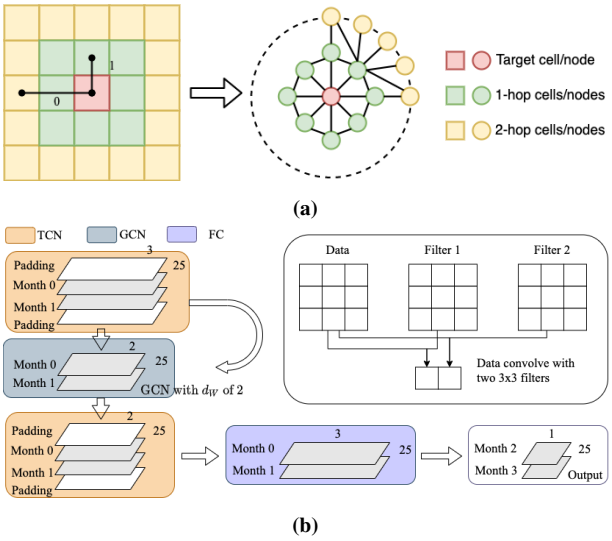

Figure 4: Demonstration to the workflow of the proposed framework. (a) Convert geographic locations in PRIO-GRID form to graph. For a certain target node at time t, any geographic areas that is directly adjacent to it is assigned an edge to the target node, with edge weight of 1; otherwise, no edge is connected, or connected with edge weight of 0. (b) The workflow of STGCN-TCN after having the formulated graph.

2.4 Demonstration of STGCN

Here we demonstrate our proposed model with a simplified example. Suppose that there is an area with 5×5 PRIO-GRID cells, for each of the grid cell we have 3 dimensional features, and we have a total of 4 steps of data available. Also, we want the model to have the ability of predicting two steps ahead. Therefore for the training, we have 2 steps of features (X_0 and X_1) and 2 steps of labels (y_2 and y_3) available.

We demonstrate the overall workflow of STGCN-TCN in Fig 4. Each module in STGCN-TCN is composed by TCN and GCN structures respectively. Here layers of TCN and GCN models are concatenated together, and extract different types of the information in turns. In this specific example, the raw features Notice that in this "sandwich" structure, we have two TCN layers and one GCN layer. Since the filter size in this example is 3, therefore, we can aggregate information upto $t-2$ and $t+2$ to representations at time t, and consider 1-hop away information using a single TCN-GCN-TCN module.

2.5 Model Learning

We use Mean Squared Error (MSE) as the training objective function to train our proposed STGCN-TCN model. This objective function represents the average squared differences between predictions for each grid cell of each step to the ground truth. Mathematically, this loss function is defined as:

$$L = \frac{1}{QN} \sum_{i=1}^{Q} \sum_{j=1}^{N} \left(\hat{y}_j^{(t+i)} - y_j^{(t+i)} \right)^2 \tag{4}$$

3 EXPERIMENT

3.1 Datasets

Two real-world datasets are used in this paper to evaluate the performance of our proposed methods against baselines on two major applications: political science and pandemic research. The statistics of these two datasets are shown in Tab. 1.

Dataset	#Nodes	#Edges	#Time Steps
ViEWS	10,677	41,711	240
Covid-19 Spread	3,221	9,473	344

Table 1: Statistics of datasets

- **ViEWS**[1] This political science dataset contains PRIO-Grid level [18] spatio-temporal conflict data concentrating on Africa, which includes records on monthly basis such as number of fatalities and geographical information. The graph at each time step is constructed follows the demonstration in Figure. 4a. There are a total number of 10,677 grid cells for Africa, and in this study we use data covering a period of 20 years, from January, 2000 to December, 2019.
- **Covid-19 Spread**[2] This pandemic research dataset contains the number of confirmed cases each day for all counties in the U.S. during 2020. Additionally, county level climate data, such as daily average/min/max temperatures, are included in this datasets as additional features. We construct the graph using the county adjacency data from U.S. Census Bureau[3]. Overall the constructed graphs contrains 3,221 counties for the U.S., and the days covered in the dataset are between Jan 22, 2020 and Dec 31, 2020.

3.2 Baselines

We compare the our proposed framework with other SOTA data-driven forecasting methods and a variant of our proposed method. These baselines are described as follows:

- **ARIMA** [2] This is a statistical based method, and is widely used for time series forecasting, and the dependent variable is predicted using only its past records.
- **Random Forest Regression** [6] This is a machine learning framework, which is composed by constructing a multitude of decision trees at the training time and take mean prediction of individual trees during the inference phase.
- **STGCN-LSTM** This is a variant of our proposed method in this paper. Instead of using TCN to model the temporal dependencies along the sequences, we use long-short term memory networks here.

3.3 Experiment Setup

ViEWS dataset contains 20 years of monthly data. We use data between Jan, 2000 and Dec 2016 as training set, and the 4 years of records are used as test set. After splitting the dataset, sequences of samples are generated by sliding a window of width $P+Q$. For this

[1] https://pcr.uu.se/research/views/
[2] https://usafacts.org/visualizations/coronavirus-covid-19-spread-map/
[3] https://www.census.gov/geographies/reference-files/2010/geo/county-adjacency.html

Dataset	ViEWS ($\times 10^{-2}$)						Covid-19 Spread, Cases						Covid-19, Death					
Time Step	1	2	3	4	5	6	1	2	3	4	5	6	1	2	3	4	5	6
ARIMA	2.96	3.05	3.07	3.12	3.16	3.23	11.01	**11.29**	11.56	11.78	11.75	11.98	9.91	10.15	10.13	10.37	10.46	**10.45**
Random Forest Regression	2.86	2.93	2.97	2.96	2.96	3.00	11.26	11.95	12.16	12.06	12.21	12.25	**9.81**	10.27	10.24	10.43	10.62	10.68
STGCN -LSTM	**2.59**	**2.31**	**2.59**	3.00	3.07	3.07	11.14	11.35	11.89	12.32	11.83	12.04	9.89	**10.13**	10.31	10.63	10.52	10.48
STGCN -TCN	2.62	2.43	2.67	**2.29**	**2.26**	**2.51**	**10.96**	11.39	**11.57**	**11.50**	**11.61**	**11.56**	9.94	10.35	**10.21**	**10.43**	**10.42**	10.52

Table 2: Comparisons of MSE results between our proposed methods and baseline methods.

dataset, we define $P = 48$ and $Q = 6$, which means that the first 48 months in the sample sequence are treated as input, while the rest 6 month are considered as ground truth. Also, we follows Hegre et al. [8] to use the logarithm of the fatalities numbers as the prediction signal.

For the Covid-19 spread dataset, it contains 344 days of daily records about No. of confirmed and death cases in U.S. For this dataset, we use data between Jan 22, 2020 and Aug 31, 2020 as the training set, and those history between Sep 1, 2020 and Dec 31, 2020 as test set. Similarily to ViEWS dataset, sequences of samples area generated by sliding windows on Covid-19 spread dataset. Here we define $P = 30$ and $Q = 6$ respectively. In this dataset, we are interested in prediction the daily confirmed cases and daily death, and these two signals are used as the prediction signal.

Our implementation of the algorithm is using Python 3.7.6 with PyTorch 1.6.0 and DGL 0.5.2 libraries. Our experiments are conducted under a computer environment with one Intel i9-9980XE CPU @ 3.00GHz and two Nvidia Quadro RTX 8000 GPU cards. Key parameters for our proposed framework in this experiment are set as follows: No. of stacked modules in STGCN-TCN $S_{tcn} = 3$; kernal size of temporal convolution $k_{tp} = 3$; batch_size = 64; learning rate $\gamma = 0.001$, and; $\beta = 0.9$ (Adam).

3.4 Results

We use Mean Squared Error (MSE) as the main evaluation metrics of the results in Table 2. As we mentioned before, the MSE is computed for a rolling six-months forecast (so six forecasts of each time period) based on the previous week 48 months of data for the ViEWS dataset. For Covid-19 Spread dataset, the prediction of the next 6 days is based on previous 30 days of records.

From the experiment results, we can see that our proposed STGCN-TCN model outperforms the competing baselines, such as ARIMA and Random Forest Regression model by a significant margin across the two experiment datasets and three prediction tasks. For example, for the STGCN-TCN model has the smallest MSE when predicting month 6 with value of 2.26×10^{-2}. Whereas the the MSE for ARIMA, Random Forest Regression and STGCN-LSTM models are 3.16×10^{-2}, 2.96×10^{-2} and 3.07×10^{-2} respectively. We illustrate the effectiveness of our proposed method on ViEWS dataset in Figure. 5. A random month is selected from the test set. Then a graph demonstrating both ground truth and predictions are generated to validate the effectiveness of STGCN-TCN. The color coding reflects the logarithm of fatalities in each PRIO-Grid cell. Light colors indicate that there is higher number of fatalities due to state-based conflicts. In the prediction, negative predictions are set to 0 for the ease of visualization. From this figure, we can see that STGCN-TCN

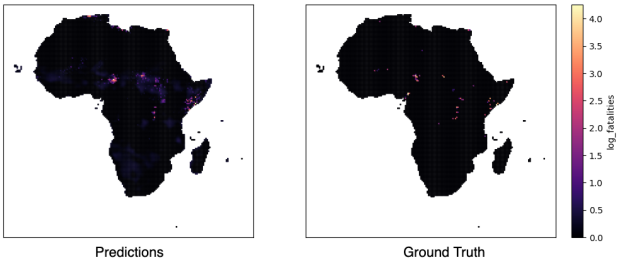

Figure 5: Visualization of prediction results using STGCN-TCN model in Africa at Sep, 2018 (ViEWS)

successfully predict those "hot spots" that may have lots of fatalities at Sep, 2018. The forecasts generated by STGCN-TCN may provide the international community early warning of conflict casualties, so conflicts could get intervened before hand, and tragedies could be avoided.

Also, from the results table, we observe that our proposed STGCN models have better prediction accuracy compared SOTA temporal only methods in most cases. This observation indicates that by introducing additional spatial dependencies into the modeling process, it is possible to generate more accurate time series forecasting predictions.

Then we compared the STGCN-TCN model with its variant STGCN-LSTM. From Figure 2, we can see that the performance of these two models are very similar for Covid-19 Spread dataset when predicting both confirmed cases and death. When it comes to the ViEWS dataset, it is very interesting to see that the STGCN-LSTM model has more accurate predictions for short-term predictions (Step 1-3), while STGCN-TCN model is predicting much better for long-term predictions. Such phenomenon is mainly due to the way that predictions are generated by LSTM and TCN. For LSTM model, the predictions are generated step-by-step, which means that the estimated value of step 2 is based on the generated estimated value of step 1. Therefore, we can clearly see that the quality of predictions generated by STGCN-LSTM is getting worse form short-term to long-term. However, since we are generating all 6 steps of predictions with fc layer in a single step for STGCN-TCN model, the observations occur to STGCN-LSTM are not the case for it anymore. That's why STGCN-TCN model performs better than its variant in generating long-term predictions across different datasets.

Method	Training Time
STGCN-TCN	45 mins
STGCN-LSTM	541 mins

Table 3: Training time of STGCN based methods on ViEWS dataset for 50 epochs.

Additionally, Table. 3 shows the training time of STGCN-based models. Having relatively close quantitative performance in generating predictions, STGCN-TCN models are much faster to train compared to its variant STGCN-LSTM model. This is because the mechanism of training temporal dependencies. One of the benefits of convolution operation is that this operator is easy to be computed in a parallel manner in GPU. That is why the training time of STGCN-LSTM model is 12 times more thatn STGCN-TCN model. Therefore, we argue that our proposed STGCN-TCN model is more cost-effective in this experiment.

4 RELATED WORK

4.1 Time-series Forecasting

This section concentrates on the descriptions of data-driven methods which consider temporal information only.

ARIMA is one of most commonly used classic methods. Even though these two methods consider the non-stationary pattern within the time series, they fail to capture the long-term dependencies within the sequence [17]. Recent deep neural networks have pushed the performance of sequence modeling to a new level. Two major types of networks are widely used. RNN and its variants, such as LSTM and GRU network have shown their great capability in modeling Natural Language Processing (NLP) related tasks [3]. Convolutional Neural Networks (CNN) are widely used in the computer vision domain, and have proven to be an effective tool for sequence modeling [21].

4.2 Graph Neural Networks

One of the advantages of representing data using graph is that the relations between data may not be necessarily in Euclidean space. GCN proposed to aggregate the neighborhood information to the target node by a first-order approximation based on graph Laplacian, and has been applied to various tasks, such as node classification, link prediction, and etc [11, 14, 15]. Other recent work targets GCN's scalability issue, and tries to address it by using local approximations to compute global node representations, such as GraphSage [7] and GAT [19].

5 CONCLUSION

In this paper, we propose a novel data-driven method, called spatio-temporal graph neural network (STGCN-TCN), to handle the time-series forecasting problem in social studies domain. By considering both spatial and temporal dynamics, and using an integrated module in the graph neural network to model both dependencies simultaneously, our proposed framework achieved SOTA performance. Accurate predictions generated by our proposed framework can provide reliable model references for social studies, such as conflict predictions or pandemic forecasting.

ACKNOWLEDGEMENT

The research reported herein was supported in part by NSF awards DMS-1737978, DGE-2039542, OAC-1828467, OAC-1931541, DGE-17236021, SMA-1539302, OAC-1828467 and DGE-1906630, ONR awards N00014-17-1-2995 and N00014-20-1-2738, Army Research Office Contract No. W911NF2110032 and IBM faculty award (Research).

REFERENCES

[1] Stefano Boccaletti, William Ditto, Gabriel Mindlin, and Abdon Atangana. 2020. Modeling and forecasting of epidemic spreading: The case of Covid-19 and beyond. *Chaos, solitons, and fractals* 135 (2020), 109794.
[2] Peter J Brockwell, Richard A Davis, and Matthew V Calder. 2002. *Introduction to time series and forecasting*. Vol. 2. Springer.
[3] Li Deng and Yang Liu. 2018. *Deep learning in natural language processing*. Springer.
[4] Yang Gao, Yi-Fan Li, Yu Lin, Hang Gao, and Latifur Khan. 2020. Deep learning on knowledge graph for recommender system: A survey. *arXiv preprint arXiv:2004.00387* (2020).
[5] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep learning*. Vol. 1. MIT press Cambridge.
[6] Ulrike Grömping. 2009. Variable importance assessment in regression: linear regression versus random forest. *The American Statistician* 63, 4 (2009), 308–319.
[7] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in neural information processing systems*. 1024–1034.
[8] Håvard Hegre, Marie Allansson, Matthias Basedau, Michael Colaresi, Mihai Croicu, Hanne Fjelde, Frederick Hoyles, Lisa Hultman, Stina Högbladh, Remco Jansen, et al. 2019. ViEWS: a political violence early-warning system. *Journal of Peace Research* 56, 2 (2019), 155–174.
[9] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.
[10] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=SJU4ayYgl
[11] Yi-Fan Li, Yang Gao, Yu Lin, Zhuoyi Wang, and Latifur Khan. 2020. Time Series Forecasting Using a Unified Spatial-Temporal Graph Convolutional Network. In *Proceedings of Preregister Workshop in 34th Conference on Neural Information Processing Systems*.
[12] Bryan Lim and Stefan Zohren. 2020. Time Series Forecasting With Deep Learning: A Survey. *arXiv preprint arXiv:2004.13408* (2020).
[13] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*.
[14] Hogun Park and Jennifer Neville. 2019. Exploiting Interaction Links for Node Classification with Deep Graph Neural Networks.. In *IJCAI*. 3223–3230.
[15] Paolo Rosso, Dingqi Yang, and Philippe Cudré-Mauroux. 2020. Beyond triplets: hyper-relational knowledge graph embedding for link prediction. In *Proceedings of The Web Conference 2020*. 1885–1896.
[16] Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing* 45, 11 (1997), 2673–2681.
[17] Sima Siami-Namini, Neda Tavakoli, and Akbar Siami Namin. 2018. A comparison of ARIMA and LSTM in forecasting time series. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 1394–1401.
[18] Andreas Forø Tollefsen, Håvard Strand, and Halvard Buhaug. 2012. PRIO-GRID: A unified spatial data structure. *Journal of Peace Research* 49, 2 (2012), 363–374.
[19] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).
[20] Billy M Williams and Lester A Hoel. 2003. Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results. *Journal of transportation engineering* 129, 6 (2003), 664–672.
[21] Sijie Yan, Yuanjun Xiong, and Dahua Lin. 2018. Spatial temporal graph convolutional networks for skeleton-based action recognition. *arXiv preprint arXiv:1801.07455* (2018).
[22] Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2018. Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, Jérôme Lang (Ed.). ijcai.org, 3634–3640. https://doi.org/10.24963/ijcai.2018/505