Deep Spatio-Temporal Encoding: Achieving Higher Accuracy by Aligning with External Real-World Data

Chen Jiang[1,*], Wenlu Wang[2], Jingjing Li[3], Naiqing Pan[1] and Wei-Shinn Ku[1]

[1] *Auburn University, Auburn, AL, USA*
[2] *Texas A&M University-Corpus Christi, Corpus Christi, TX, USA*
[3] *Meta, Menlo Park, CA, USA*

Abstract

Spatio-temporal deep learning has drawn a lot of attention since many downstream real-world applications can benefit from accurate predictions. For example, accurate prediction of heavy rainfall events is essential for effective urban water usage, flooding warning, and mitigation. In this paper, we propose a strategy to leverage spatially connected real-world features to enhance prediction accuracy. Specifically, we leverage spatially connected real-world climate data to predict heavy rainfall risks in a broad range in our case study. We experimentally ascertain that our Trans-Graph Convolutional Network (TGCN) accurately predicts heavy rainfall risks and real estate trends, demonstrating the advantage of incorporating external spatially-connected real-world data to improve model performance, and it shows that this proposed study has a significant potential to enhance spatio-temporal prediction accuracy, aiding in efficient urban water usage, flooding risk warning, and fair housing in real estate.

Keywords

Spatial-temporal Analysis, Deep Learning, Transformer,

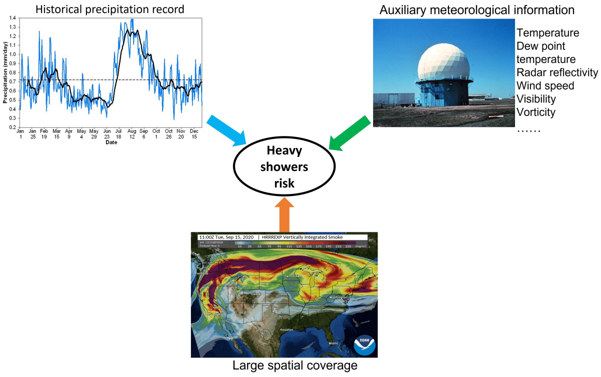

Figure 1: An example of spatial and temporal features in the case study of precipitation prediction.

1. Introduction

Spatio-temporal predictions have been extensively studied due to their impact on real-world applications [1, 2, 3, 4, 5]. For example, heavy rainfall events can cause significant damage to infrastructure and pose serious threats to human safety. Predicting these events with greater accuracy allows better preparation and response [6], ultimately saving lives and reducing the economic impact of such events.

Deep learning methods, such as deep spatio-temporal prediction models [7, 8], have improved the performance of rainfall forecasting over the years. However, the role of external data in enhancing the prediction accuracy is still controversial. Some argue that external data can provide more useful information for the prediction model, while others claim that external data can introduce more noise and complexity to the learning process. In this study, we propose to improve spatio-temporal predictions by combining spatially-linked external real-world data along with a TGCN to learn the spatio-temporal dependencies from the combined data. As it has been proven that utilizing more multi-source real-world data is more likely to lead to higher accuracy [9], our study aims to introduce a fresh perspective on integrating external real-world data into the proposed framework. We use heavy rainfall prediction as a case study for our proposed method, and overall we aim to provide accurate spatio-temporal predictions by leveraging as much information as possible, enabling better decision-making for a broad range of spatio-temporal applications and at the same time offering a novel angle and a comprehensive evaluation to demonstrate the feasibility of integrating additional external real-world data without the necessity of customizing transformer attention mechanisms. Our approach is experimentally validated by predicting heavy rainfall events and real estate hotspots.

The traditional method for predicting heavy rainfall involves manually engineering features from weather data, including temperature, pressure, humidity, etc. Meteorologists rely on their expertise to interpret this data and forecast future weather patterns. This process entails observing and analyzing atmospheric factors to predict weather patterns. However, this traditional approach is time-consuming, labor-intensive, and susceptible to human error, especially when dealing with large datasets. As data grows, it becomes increasingly challenging to analyze large amounts of information by hand.

Previous research has investigated using deep learning for precipitation prediction [10, 11] with promising results. However, some limitations can be significantly improved to enhance deep model performance. One area with room for enhancement is leveraging spatial dependencies. To tackle this challenge, we propose a model that integrates both Graph Convolution Networks (GCNs) and a Transformer. This model enables combining external spatially-linked data for spatio-temporal predictions.

Specifically, we employ a GCN to analyze the adjacency matrix on a grid level and generate correlations between each grid element. The GCN captures the spatial relationships and dependencies among neighboring grid points, allowing for a comprehensive understanding of the data's

Published in the Proceedings of the Workshops of the EDBT/ICDT 2024 Joint Conference (March 25-28, 2024), Paestum, Italy

[*] Corresponding author.

czj0042@auburn.edu (C. Jiang); wenlu.wang@tamucc.edu (W. Wang); jingjingli@meta.com (J. Li); nzp0030@auburn.edu (N. Pan); weishinn@auburn.edu (W. Ku)
0009-0000-6888-6643 (C. Jiang); 0000-0002-4829-1068 (W. Wang); 0000-0002-1465-7738 (N. Pan); 0000-0001-8636-4689 (W. Ku)

spatial dynamics. We then utilize a Transformer model to encode the temporal precipitation data and combine it with the spatial correlations obtained from the GCNs. By combining the GCNs and the Transformer within the proposed TGCN model, we create a framework that harnesses both the spatial and temporal dimensions of the data.

2. Related Work

2.1. Graph Neural Networks

Graph Convolutional Networks (GCNs) are a type of deep learning model designed to process data represented in a graph structure, such as social or sensor networks [12]. GCNs have demonstrated their effectiveness in various applications, including node classification, link prediction, and recommendation systems [13, 14, 15, 16]. The concept of Graph Neural Networks (GNNs) was initially introduced in [17] and further expanded upon in subsequent research by [18]. GNNs, a type of recurrent neural network (RNN), iteratively propagate information from neighboring nodes until reaching a stable fixed point. This iterative process has traditionally been computationally expensive, but recent studies, such as [19], have made significant improvements in this area. Inspired by the success of Convolutional Neural Networks (CNNs) in computer vision, which extract high-level features from images using convolution and pooling layers, current models aim to adapt these layers to directly process graph inputs. GCNs can be categorized into two types of graph convolution layers: spectral graph convolution and localized graph convolution, as discussed in [20]. Early research primarily focused on spectral graph convolutions, pioneered by [21]. The current state-of-the-art model, GCN, further simplified the graph convolution operation by employing a localized first-order approximation. However, spectral methods require operations on the entire graph Laplacian during training, which can be computationally expensive. Several subsequent works, such as FastGCN [22] have aimed to alleviate this issue.
Recently, researchers have explored the application of GCNs in time series prediction. For example, spatio-temporal GCN-based approaches have been proposed for traffic flow prediction [23], and the integration of time-aware topological information into GCNs using the mathematical framework of zigzag persistence [24].

2.2. Spatial Temporal Prediction

In this section, we discuss various existing temporal and spatial-temporal forecasting methods. For example, Recurrent Neural Networks (RNNs), especially long-short-term memory (LSTM) [25], have gained popularity in time series forecasting [26]. Convolutional Neural Networks (CNN) and its variant Temporal Convolutional Neural Networks (TCN) are another option for sequence prediction [27], offering parallel computations compared to RNNs [28]. In recent years, researchers have explored Transformers and its variants in time series forecasting, achieving state-of-the-art performance in tasks like energy consumption and stock market [29, 30, 31]. Designing a model capable of comprehensively capturing both spatial and temporal patterns represents another emerging trend in spatial-temporal prediction tasks [32, 33]. For example, [33] introduced a spatial-temporal graph neural network for predicting traffic flow.

3. Methodology

In this section, we detail our model architecture and the benefits of our design.

3.1. Overview

The architecture we propose, illustrated in Figure 2, incorporates a combination of techniques to enhance the prediction model. We begin by utilizing a transformer encoder to effectively encode the time series precipitation data, and then integrate local climate features into the model, enabling a comprehensive understanding of the factors influencing heavy rainfall.

To address spatial dependencies and relationships among grid points, a GCN is introduced. This GCN learns the spatial dependencies within the dataset, considering the interconnectedness of grids based on their spatial locations. By leveraging the GCN, the model becomes capable of capturing and integrating spatial information, thereby enhancing prediction accuracy.

The latent code, which combines the encoded time series precipitation data and the spatially connected local climate features learned through the GCN, is fed into a multi-layer perceptron (MLP) for prediction. This integrated architecture allows the MLP model to leverage the fused information, including temporal precipitation data, other climate features, and spatial factors, to effectively learn and infer future heavy rainfall areas.

3.2. Model Architecture

3.2.1. Preliminaries

Our proposed TGCN model consists of **Encoder**, **GCNs** and **Multi-layer Perceptron** (MLP) layers. The major component in the transformer is the **Multi-head self-attention**.

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \qquad (1)$$

Where the K and V are matrices that store the keys and values. Q is the query that will map against a set of keys.

3.2.2. Transformer-based Encoder

We have developed a predictive model using the Transformer architecture, tailored for heavy rainfall forecasts. Unlike traditional methods that only use past rainfall data, our model factors in numerous external variables to boost accuracy. We examine local features, including geography, atmospheric conditions (pressure, temperature, wind), humidity, and topography, all of which influence heavy rainfall likelihood in a specific area. Therefore, we have developed a transformer-based prediction model [34] that incorporates GCNs to process the spatial features. By doing so, our model can capture the spatial relationships among various features in a graph structure, such as the dependencies between grid point locations and their corresponding climate data. The integration of the GCNs enhances our model's ability to capture both temporal and spatial information. Our model design starts with a transformer encoder capturing temporal precipitation patterns, followed by embedding this data and merging it with local climate data like moisture and

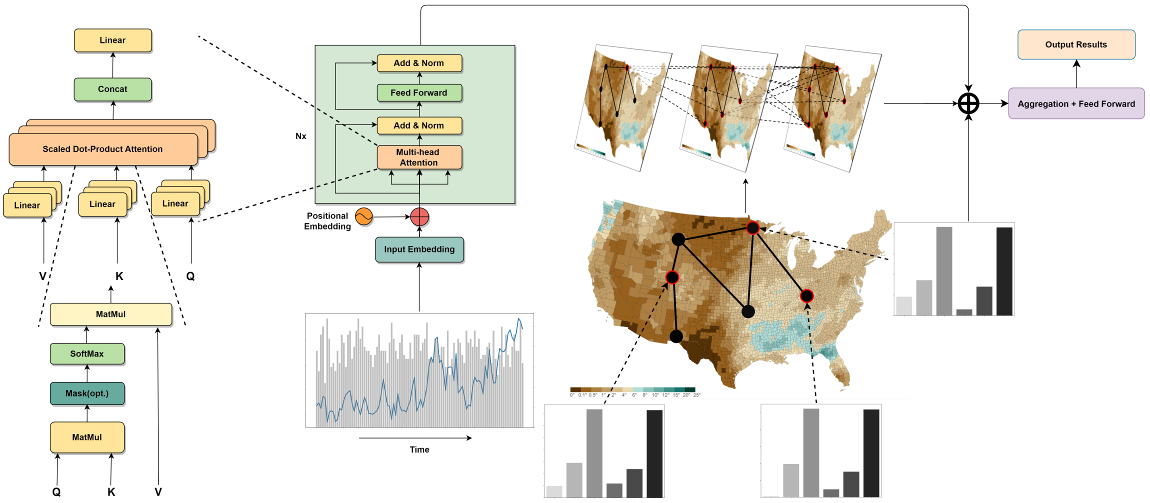

Figure 2: Design Flow of the Trans-Graph Convolutional Prediction Model: The Trans-Graph Convolutional Prediction Model incorporates a transformer layer for time-series precipitation data, a GCN for local climate features and spatial relationships among grid points, and a four-layer MLP model for the final prediction.

humidity. We enhance prediction accuracy with this added context.

3.2.3. Graph Convolutional Networks

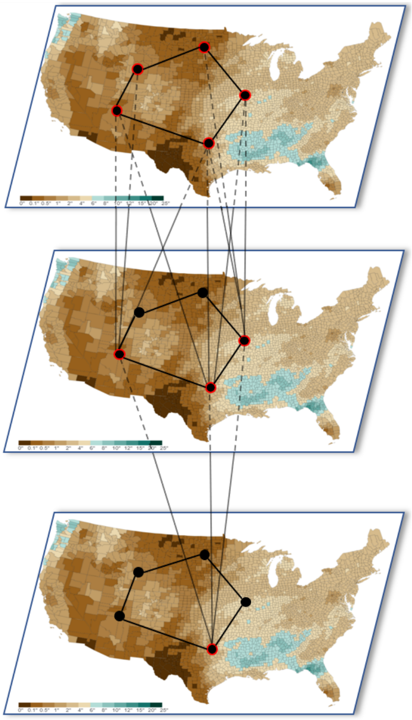

Figure 3: Graph Convolutional Network Architecture: The input data consists of the spatial relation matrix and spatially connected climate data. The nodes in the figure are for illustrative purposes.

GCNs have received considerable attention in recent years and have shown impressive performance in various applications. In this study, we aim to improve the performance of our model by integrating a GCN on top of a Transformer encoder model. The GCN model is specifically designed to capture the spatial relationships between each node in the graph and enhance the overall representation of the input data.

As illustrated in Figure 3, GCNs involve learning a linear transformation of the feature vectors of each node in a graph, which is then used to update the node features by aggregating information from the node's neighbors. Mathematically, this can be expressed as:

$$h_{v_i}^{(l+1)} = \sigma\left(\sum_{v_j \in \mathcal{N}(v_i)} \frac{1}{c_{ij}} W^{(l+1)} h_{v_j}^{(l+1)}\right) \qquad (2)$$

In the equation, $h_{v_i}^{(l+1)}$ represents the feature vector of node v_i at layer $l+1$, $W^{(l+1)}$ denotes the learnable weight matrix for layer $l+1$, $\mathcal{N}(v_i)$ represents the set of neighbors of node v_i, and c_{ij} is a normalization constant that ensures proper scaling of the aggregated information. The function σ denotes a non-linear activation function, which introduces non-linearity into the model. In our specific case, we utilize the ReLU activation function. This equation can be interpreted as calculating a weighted sum of the feature vectors of the neighbors of node v_i at layer $l+1$, where the weights are determined by the learned weight matrix $W^{(l+1)}$. Then, a non-linear activation function is applied to obtain the updated feature vector $h_{v_i}^{(l+1)}$ for node i at layer $l+1$. This process is repeated across multiple layers to learn expressive representations of the graph data.

For the final prediction, we utilize a four-layer MLP model that combines time series data with other features, effectively leveraging both temporal and spatial information captured by our model for more accurate predictions.

By leveraging the transformer architecture, incorporating GCNs, and utilizing a four-layer MLP model, our approach enables the effective integration of temporal and spatial information for improved prediction accuracy.

3.2.4. Jointly Learning

As illustrated in Figure 2, we propose to map temporal data and non-temporal data into the same latent space and merge the latent vectors for the subsequent prediction task.

To encode the local climate features and capture the spatial dependencies among the grid points for data x_c, we employ a GCN to learn the relationships and dependencies within the spatial domain. The output hidden features at a specific layer L can be denoted as $h_c^{(L)}$. Equation 2 is applied in this context. Assuming we use L^c layers in total, and we use the final layer to summarize climate information, which is defined as:

$$\mathbf{h_c} = h_v^{(L^c)} \tag{3}$$

where $h_v^{(0)} = x_c$

In this equation, h_c represents the hidden features at layer L, which are obtained by applying the ReLU activation function to the sum of the weighted input features $W_c^{(L)} h_c^{(L-1)}$ and the bias term $b_c^{(L)}$ from Equation 2.

We encode temporal precipitation data using a transformer encoder [34],

$$\mathbf{h_t} = TransformerEncoder(x_t) \tag{4}$$

$$\mathbf{h_t} \in \mathbf{R}^{d_t} \tag{5}$$

. Since x_t and x_c are encoded as $\mathbf{h_t}$ and $\mathbf{h_c}$, we define the merged hidden state as $\mathbf{h_m}$

$$\mathbf{h_m} = CONCAT(\mathbf{h_t}, \mathbf{h_c}) \tag{6}$$

To further process the merged information, we use another multi-layer perceptron specifically trained for the prediction task. Similarly, we define the l-th layer network as (assuming L^n layers in total)

$$h_n^{(l)} = ReLu(W_n^{(l)} h_n^{(l-1)} + b_n^{(l)}) \tag{7}$$

where $h_n^{(0)} = \mathbf{h_m}$, and $h_n^{(l-1)}$ is the input of the (*l-1*)-th layer in the *i*-th position. $W_n^{(l)}$ and $b_n^{(l)}$ are model parameters.

We use the output from the last layer for prediction

$$\bar{y} = sigmoid(h_n^{(L^n)}) \tag{8}$$

Loss is measured with the Binary Cross-Entropy loss (BCE)

$$loss = BCEloss(\bar{y}, y) \tag{9}$$

The binary cross entropy (BCE) loss can be formulated as follows:

$$BCEloss = -\frac{1}{N} \sum_{i=1}^{N} [y_i \log(p_i) + (1 - y_i) \log(1 - p_i)] \tag{10}$$

where: N is the total number of samples, y_i is the true label for sample i, p_i is the predicted probability i, log denotes the natural logarithm.

4. Experimental Validation

4.1. Datasets

Our data and code are publicly available[1]. In our dataset, the train and test split ratio is 7:3.

4.1.1. Precipitation Dataset

Our precipitation dataset is sourced from the NOAA HRRR dataset[2], offering real-time climate data at a 3 km spatial resolution and 1-hour temporal resolution. This dataset [35] encompasses total precipitation, precipitation rate, and nine additional climate variables, including humidity (%), moisture availability (%), pressure (Pa), wind speed (m/s), and total cloud cover (%). Simulated brightness temperature data is acquired from the GOES 11 satellite [3]. The precipitation data consist of the following three types:

- Temporal precipitation data, denoted as x_t, as shown in Table 1 and Figure 5. It captures the historical patterns and fluctuations in precipitation over time. Specifically, we define the temporal precipitation rate and total accumulated precipitation over the past 6 hours as x_h, which consists of N timestamps:

  $$x_t = \{x_t^1, x_t^2, ..., x_t^N\}$$

  $x_t^i{}_{i \in \{1..N\}}$ represents the average price for the i-th timestamp.
- Local climate data x_c: The dataset comprises twelve local climate variables, including temperature, humidity, wind speed, atmospheric pressure, and various other meteorological factors.
- Spatial location data x_s: Each grid point in the dataset represents a specific location within the study area, such as a region or a cell. To represent the relationships between these grid points, we used an adjacency matrix. In the adjacency matrix, a value of 0 indicates that two grid points are not neighbors, while a value of 1 denotes their neighboring relationship.

4.1.2. Real-estate Dataset

The real estate dataset captures the dynamics of the U.S. real estate market by collecting spatially correlated data from multiple sources. It consists of 7,436 neighborhoods, 567 cities, 304 counties, 225 metros, and 50 states across the U.S. The data are connected through spatial locations, forming a multi-level spatial hierarchy. The dataset consists of three main components: census data, pricing history, and school district information. Here are some statistics about the real estate dataset:

- Spatial Hierarchy Levels: The dataset includes a multi-level spatial hierarchy, including information at the state, metro, county, city, and neighborhood levels.
- Census Data: The census data consists of 16 variables related to various aspects of housing prices, personal income, demographics, and spatial information.
- Pricing History: The dataset includes temporal housing price history for each neighborhood, spanning from 1996 to 2019.
- School District Information: The dataset incorporates school district information. It provides details on the number of school districts present in each county within the studied area. Additionally, the dataset includes information on the top school district(s) within the region.

[1] https://github.com/jiang28/Deep-Spatio-Temporal-Encoding

[2] https://rapidrefresh.noaa.gov/hrrr/

[3] https://www.goes.noaa.gov/

GridID	Longitude	Latitude	Grid Points	Grid Spacing	Vertical Level
1	122.71	21.13	1799 × 1059	3 km	50

Time Stamps	2022/09/23 00:00	2022/09/23 01:00	2022/09/23 02:00	...	2022/10/02 00:00
Precipitation rate (mm/hour)	0.0	0.72	0.94	...	0
Total Precipitation (mm)	0.01	1.88	4.3	...	31.61

Table 1
Temporal data format. It has data on the grid id, longitude, latitude, grid points, grid spacing, vertical level, timestamps, total precipitation, and precipitation rate.

To facilitate the task of predicting real estate hotspots, the dataset is classified into two classes based on the house price increase rate for each neighorhood: 1 for hotspots and 0 for non-hotspots. The detailed settings of the Real-estate Dataset can be found in [36].

4.2. Evaluation Metrics

We evaluate the performance of a classification system using various metrics, including Accuracy, Recall, Precision, F1-score, and ROC. These metrics are calculated based on the number of true positives (t_p), false positives (f_p), false negatives (f_n), and true negatives (t_n). Accuracy measures the proportion of observations, both positive and negative, that were correctly classified by the system, and can be computed using the formula:

$$acc = \frac{t_p + t_n}{t_p + f_p + t_n + f_n}$$

Recall measures the proportion of true positives that were correctly identified by the system, and can be computed using the formula:

$$recall = \frac{t_p}{t_p + f_n}$$

Precision measures the proportion of identified positives that were actually true positives, and can be computed using the formula:

$$precision = \frac{t_p}{t_p + f_p}$$

F1-score is a weighted average of precision and recall, and provides a single measure of the system's accuracy on the dataset, and can be computed using the formula:

$$F1 = 2 * \frac{precision * recall}{precision + recall}$$

ROC (Receiver Operating Characteristic) curve is a graphical plot that illustrates the performance of a binary classifier system. It is created by plotting the True Positive Rate (TPR) against the False Positive Rate (FPR), which can be computed using the formulas:

$$TPR = \frac{t_p}{t_p + f_n}$$

$$FPR = \frac{f_p}{f_p + t_n}$$

Overall, these metrics provide a comprehensive evaluation of a classification system's performance and can help identify areas for improvement.

4.3. Heavy Rainfall Prediction

Study Area: Figure 4 presents the location of the study area in this study. It consists of 10,000 grids across the state of Florida in the U.S.

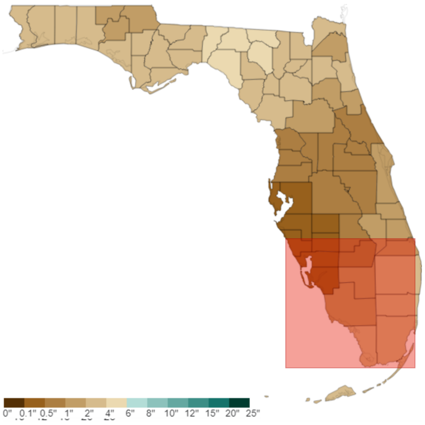

Figure 4: The study area consists of 10,000 grids across South Florida in the United States. The figure shows the observed precipitation values in each county within this area.

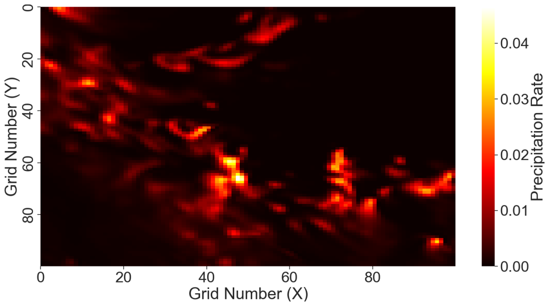

Figure 5: Study Area Precipitation Rate Heatmap: 100x100 grid region on September 28, 2022, at 13:00 (mm/s).

Our study identifies heavy rainfall risk areas based on precipitation rate. Following the United States Geological Survey (USGS) standard[4], we define the heavy rainfall risk as follows:

$$\text{Class} = \begin{cases} 0, & \text{if } R < 4 \text{ mm/hr} \\ 1, & \text{if } R \geq 4 \text{ mm/hr} \end{cases}$$

Recognizing the significance of precipitation rate as a critical factor, our objective is to pinpoint areas that are susceptible to encountering heavy rainfall within the next hour. The classification into two classes simplifies the problem and provides a clear distinction between areas with different levels of heavy rainfall risk. Using a 4 mm/hour

[4] https://www.usgs.gov/

Model	Accuracy	Precision		Recall		F1-score		ROC
		0	1	0	1	0	1	
RF	81%	0.79	0.84	0.79	0.81	0.87	0.81	0.813
SVM	77.2%	0.77	0.78	0.76	0.78	0.77	0.78	0.772
DT	76.5%	0.73	0.77	0.77	0.73	0.75	0.75	0.754
LR	90%	0.91	0.90	0.90	0.91	0.91	0.90	0.904
MLP	87.8%	0.82	0.94	0.93	0.84	0.87	0.89	0.879
LSTM	86.6%	0.79	0.96	0.96	0.79	0.86	0.87	0.874
Transformer	93.5%	0.88	0.98	0.98	0.90	0.93	0.94	0.941
TGCN (*Ours*)	**95.6%**	**0.93**	**0.97**	**0.97**	**0.94**	**0.95**	**0.96**	**0.954**

Table 2
When comparing model performance on the Real Estate dataset, the proposed model has achieved an accuracy of 95.6%.

Model	Accuracy	Precision		Recall		F1-score		ROC
		0	1	0	1	0	1	
RF	74.4%	0.65	0.78	0.56	0.84	0.60	0.81	0.701
SVM	54.1%	0.28	0.63	0.20	0.73	0.23	0.67	0.461
DT	80.5%	0.91	0.74	0.69	0.93	0.78	0.82	0.807
LR	78.8%	0.85	0.74	0.70	0.87	0.77	0.80	0.87
MLP	80.3%	0.83	0.78	0.77	0.84	0.80	0.81	0.804
LSTM	83.1%	0.87	0.80	0.79	0.88	0.83	0.84	0.832
Transformer	83.4%	0.85	0.82	0.82	0.85	0.83	0.83	0.835
TGCN (*Ours*)	**86.6%**	**0.90**	**0.83**	**0.82**	**0.91**	**0.86**	**0.87**	**0.867**

Table 3
When comparing model performance on the Precipitation dataset, the proposed model has achieved an accuracy of 86.6%.

threshold, we classify areas as either low-risk (labeled as 0) or high-risk (labeled as 1). For example, out of 10,000 grid points in the study area, 4,798 have a potential for heavy rain risk, while 5,202 do not. This classification simplifies decision-making and resource allocation.

4.4. Baselines

We use the following baseline methods:

- Random Forest (RF) [37]
- Support Vector Machine (SVM) [38]
- Decision Tree (DT) [39]
- Linear Regression (LR) [40]
- Multilayer Perceptron (MLP) [41]
- Long Short Term Memory (LSTM) [25]
- Transformer [34]

5. Performance Analysis

Based on the results presented in Table 2 and Table 3, we can analyze the performance of different models on the Real Estate dataset and the Precipitation dataset, respectively.

In Table 2, the proposed model outperforms all the baseline models with an accuracy of 95.6%. The proposed model also exhibits the highest precision for both classes (0 and 1), achieving 0.93 and 0.97, respectively. It demonstrates high recall values for both classes as well. The F1 scores are also higher for the proposed model compared to the baseline models, indicating a better balance between precision and recall. The TGCN model's performance is further reflected in the ROC score of 0.954, which indicates its ability to discriminate between the two classes effectively.

Table 3 shows that the proposed model again achieves the highest accuracy of 86.6%. Similar to the Real Estate dataset, the TGCN model demonstrates superior precision and recall values for both classes compared to the baseline models. It achieves precision scores of 0.9 and 0.83 for classes 0 and 1, respectively, along with recall scores of 0.82 for class 0 and 0.85 for class 1. The F1 scores also indicate the TGCN model's overall better performance. The ROC score for the TGCN model is 0.867.

These results demonstrate that the proposed TGCN model consistently outperforms the other models on both datasets in terms of accuracy, precision, recall, F1 score, and ROC score. The TGCN model's ability to capture temporal, non-temporal, and spatial information through its integration of the transformer layer and the graph convolutional network contributes to its good performance in identifying and predicting hotspots and heavy rainfall areas.

6. Conclusion

In conclusion, the accurate prediction of heavy rainfall events is crucial for effective urban water usage, disaster response, and mitigation efforts. This paper proposed a prediction model that leverages spatially connected features and real-world climate data to predict heavy rainfall risks across a broad range. Through extensive experimentation, it was observed that the TGCN model outperformed the other machine learning methods in forecasting both heavy rainfall events and real estate trends.

7. Future Work and Limitations

While this study successfully demonstrated the effectiveness of the proposed TGCN model in predicting heavy rainfall risks, there are several avenues for future research and improvement.

We plan to incorporate more diverse and comprehensive datasets, including additional meteorological and geographical features. This expansion has the potential to enhance the accuracy and generalizability of the TGCN model. Furthermore, we are considering the integration of real-time data streams and the utilization of advanced data fusion techniques to further enhance the model's forecasting capabilities.

Acknowledgement

This work was partially supported by the National Science Foundation (NSF) under Grant No. 2318641. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of the National Science Foundation.

References

[1] A.-T. Kuo, H. Chen, W.-S. Ku, Bert-trip: Effective and scalable trip representation using attentive contrast learning, in: 2023 IEEE 39th International Conference on Data Engineering (ICDE), IEEE Computer Society, 2023, pp. 612–623.

[2] P.-Y. Ting, T. Wada, Y.-L. Chiu, M.-T. Sun, K. Sakai, W.-S. Ku, A. A.-K. Jeng, J.-S. Hwu, Freeway travel time prediction using deep hybrid model–taking sun yat-sen freeway as an example, IEEE Transactions on Vehicular Technology 69 (2020) 8257–8266.

[3] A. Datta, S. Banerjee, A. O. Finley, A. E. Gelfand, Hierarchical nearest-neighbor gaussian process models for large geostatistical datasets, Journal of the American Statistical Association 111 (2016) 800–812.

[4] B. Gräler, E. J. Pebesma, G. B. Heuvelink, Spatio-temporal interpolation using gstat., R J. 8 (2016) 204.

[5] Z. Diao, X. Wang, D. Zhang, Y. Liu, K. Xie, S. He, Dynamic spatial-temporal graph convolutional neural networks for traffic forecasting, in: Proceedings of the AAAI conference on artificial intelligence, volume 33, 2019, pp. 890–897.

[6] K. Kitchat, M.-H. Lin, H.-S. Chen, M.-T. Sun, K. Sakai, W.-S. Ku, T. Surasak, A deep reinforcement learning system for the allocation of epidemic prevention materials based on ddpg, Expert Systems with Applications 242 (2024) 122763.

[7] F. Amato, F. Guignard, S. Robert, M. Kanevski, A novel framework for spatio-temporal prediction of environmental data using deep learning, Scientific reports 10 (2020) 22243.

[8] H. Liu, X. Mi, Y. Li, Smart deep learning based wind speed prediction model using wavelet packet decomposition, convolutional neural network and convolutional long short term memory network, Energy Conversion and Management 166 (2018) 120–131.

[9] K. Bi, L. Xie, H. Zhang, X. Chen, X. Gu, Q. Tian, Accurate medium-range global weather forecasting with 3d neural networks, Nature (2023) 1–6.

[10] A. Moraux, S. Dewitte, B. Cornelis, A. Munteanu, A deep learning multimodal method for precipitation estimation, Remote Sensing 13 (2021) 3278.

[11] X. Shi, Z. Gao, L. Lausen, H. Wang, D.-Y. Yeung, W.-k. Wong, W.-c. Woo, Deep learning for precipitation nowcasting: A benchmark and a new model, Advances in neural information processing systems 30 (2017).

[12] T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, arXiv preprint arXiv:1609.02907 (2016).

[13] L. Cai, B. Yan, G. Mai, K. Janowicz, R. Zhu, Transgcn: Coupling transformation assumptions with graph convolutional networks for link prediction, in: Proceedings of the 10th international conference on knowledge capture, 2019, pp. 131–138.

[14] L. Gan, X. Yang, N. Narisetty, F. Liang, Bayesian joint estimation of multiple graphical models, Advances in Neural Information Processing Systems 32 (2019).

[15] H. Gao, Z. Wang, S. Ji, Large-scale learnable graph convolutional networks, in: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, 2018, pp. 1416–1424.

[16] H. Wang, M. Zhao, X. Xie, W. Li, M. Guo, Knowledge graph convolutional networks for recommender systems, in: The world wide web conference, 2019, pp. 3307–3313.

[17] M. Gori, G. Monfardini, F. Scarselli, A new model for learning in graph domains, in: IEEE International Joint Conference on Neural Networks, volume 2, IEEE, 2005, pp. 729–734.

[18] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, G. Monfardini, The graph neural network model, IEEE Trans. Neural Networks 20 (2009) 61–80.

[19] Y. Li, D. Tarlow, M. Brockschmidt, R. S. Zemel, Gated graph sequence neural networks, in: ICLR, 2016.

[20] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, J. Leskovec, Graph convolutional neural networks for web-scale recommender systems, in: SIGKDD, ACM, 2018, pp. 974–983.

[21] J. Bruna, W. Zaremba, A. Szlam, Y. LeCun, Spectral networks and locally connected networks on graphs, in: ICLR, 2014.

[22] J. Chen, T. Ma, C. Xiao, Fastgcn: Fast learning with graph convolutional networks via importance sampling, in: ICLR, OpenReview.net, 2018.

[23] B. Yu, H. Yin, Z. Zhu, Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting, arXiv preprint arXiv:1709.04875 (2017).

[24] S. Guo, Y. Lin, N. Feng, C. Song, H. Wan, Attention based spatial-temporal graph convolutional networks for traffic flow forecasting, in: Proceedings of the AAAI conference on artificial intelligence, volume 33, 2019, pp. 922–929.

[25] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (1997) 1735–1780.

[26] S. McNally, J. Roche, S. Caton, Predicting the price of bitcoin using machine learning, in: 2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), IEEE, 2018, pp. 339–343.

[27] S. D. Yeddula, C. Jiang, B. Hui, W.-S. Ku, Traffic accident hotspot prediction using temporal convolutional networks: A spatio-temporal approach, in: Proceedings of the 31st ACM International Conference on Advances in Geographic Information Systems, 2023, pp. 1–4.

[28] A. Borovykh, S. Bohte, C. W. Oosterlee, Conditional time series forecasting with convolutional neural networks, arXiv preprint arXiv:1703.04691 (2017).

[29] S. Wu, X. Xiao, Q. Ding, P. Zhao, Y. Wei, J. Huang, Adversarial sparse transformer for time series forecasting, Advances in neural information processing systems 33 (2020) 17105–17115.

[30] J. Yoo, Y. Soun, Y.-c. Park, U. Kang, Accurate multivariate stock movement prediction via data-axis trans-

former with multi-level contexts, in: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 2037–2045.

[31] Y. Liu, S. Wang, J. Chen, B. Chen, X. Wang, D. Hao, L. Sun, Rice yield prediction and model interpretation based on satellite and climatic indicators using a transformer method, Remote Sensing 14 (2022) 5045.

[32] Z. Lin, M. Li, Z. Zheng, Y. Cheng, C. Yuan, Self-attention convlstm for spatiotemporal prediction, in: Proceedings of the AAAI conference on artificial intelligence, volume 34, 2020, pp. 11531–11538.

[33] X. Wang, Y. Ma, Y. Wang, W. Jin, X. Wang, J. Tang, C. Jia, J. Yu, Traffic flow prediction via spatial temporal graph neural network, in: Proceedings of the web conference 2020, 2020, pp. 1082–1092.

[34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, arXiv preprint arXiv:1706.03762 (2017).

[35] C. Jiang, W. Wang, N. Pan, W.-S. Ku, A multimodal geo dataset for high-resolution precipitation forecasting, in: Proceedings of the 31st ACM International Conference on Advances in Geographic Information Systems, 2023, pp. 1–4.

[36] C. Jiang, J. Li, W. Wang, W.-S. Ku, Modeling real estate dynamics using temporal encoding, in: Proceedings of the 29th International Conference on Advances in Geographic Information Systems, 2021, pp. 516–525.

[37] T. K. Ho, Random decision forests, in: Proceedings of 3rd international conference on document analysis and recognition, volume 1, IEEE, 1995, pp. 278–282.

[38] B. E. Boser, I. M. Guyon, V. N. Vapnik, A training algorithm for optimal margin classifiers, in: Proceedings of the fifth annual workshop on Computational learning theory, 1992, pp. 144–152.

[39] W.-Y. Loh, Classification and regression trees, Wiley interdisciplinary reviews: data mining and knowledge discovery 1 (2011) 14–23.

[40] J. A. Nelder, R. W. Wedderburn, Generalized linear models, Journal of the Royal Statistical Society: Series A (General) 135 (1972) 370–384.

[41] L. B. Almeida, C1. 2 multilayer perceptrons, Handbook of Neural Computation C 1 (1997).