Tangent Space-Free Lorentz Spatial Temporal Graph Convolution Networks

Abdelrahman Mostafa and Guoying Zhao, *Fellow, IEEE*

***Abstract*—Spatial Temporal Graph Convolution Networks (ST-GCNs) have been proposed to embed spatio-temporal graphs. However, these networks used the Euclidean space as the embedding space which does not exploit the structure of the embedded graphs. Euclidean space has been shown not to be the ideal space for embedding graphs especially with tree-like structures. In this work, we make use of hyperbolic geometry and introduce a compact tangent space-free Lorentz ST-GCN and call it LSTGCN that perform the network operations directly on the manifold without resorting to the tangent space. The network uses spatial and temporal modules to propagate features between adjacent nodes in both, the spatial domain and the temporal domain, respectively. In addition, we introduce an attention module which can automatically determine the similarity of nodes without the need for the graph adjacency matrix. Experiments have been conducted on traffic flow forecasting tasks to show the effectiveness of the proposed compact Lorentz model.**

***Index Terms*—Hyperbolic geometry, Lorentz model, spatial temporal graph convolution networks, spatio-temporal graphs, traffic forecasting.**

I. Introduction

MANY problems can be modelled using graph structures with the entities as the graph nodes and the relationship between the entities as the graph edges. For example, citation networks can be represented as static graphs [1] and the sequence of human skeletons as spatio-temporal graphs [2]. Graph Convolution Networks (GCNs) were proposed to model the spatial relationship between static graph nodes and generalized the convolution operation to any graph data [3], [4]. Spatial Temporal Graph Convolution Networks (ST-GCNs) were then introduced to embed spatio-temporal graphs, for example, for traffic forecasting [5] and for human action recognition [2], [6], [7] tasks. However, these methods used the Euclidean space for embedding graph features which is not the natural space for embedding such data and introduces a large distortion [1], [8]. The hyperbolic space is the natural space to represent graph data as the space volume is growing exponentially which is perfect specially for tree-like data that grow exponentially with respect to the tree depth [1], [8], [9]. Subsequently, Hyperbolic Graph Convolution Networks (HGCNs) were proposed to take advantage of the hyperbolic geometry to embed static graph data in the hyperbolic space [1], [10]. HGCNs were extended to embed spatio-temporal graph data [9] for the graph classification human action recognition task. A comprehensive survey about hyperbolic learning can be found in [8].

However, these methods used the tangent space which is a Euclidean local approximation to the hyperbolic space at a point to perform network operations [1], [10], [11], [12]. The work in [13] used normalization to keep the learnt features for static graphs on the manifold. Their implementation was a relaxation and the points were not restricted to lie on the manifold. In [14], the authors built a hyperbolic network by imposing the orthogonal constraint on a sub-matrix of the transformation matrix for static graphs. In [15], the authors used manifold-preserving Lorentz transformations to learn features for static graphs. Different from earlier works, we design a tangent space-free Lorentz network to embed features for spatio-temporal graphs that are more challenging than static graphs due to the graph features evolving through time and without the need for normalization. We evaluate the performance of the proposed method on the traffic forecasting task. Traffic flow prediction is an important area of research in order to build a smart traffic management systems. Deep learning methods have been used for traffic flow prediction [16], [17], [18], [19], [20], [21]. Methods based on GCNs were successfully applied to this task [5], [22], [23], [24]. However, all the methods used the Euclidean space for embedding the spatio-temporal graph features which does not match the topology of the graph. The contribution of this work can be summarized as:

- We design a compact tangent space-free Lorentz ST-GCN (LSTGCN) to embed spatio-temporal graphs which exploits both spatial and temporal connections. To the best of our knowledge, this is the first work to optimize a hyperbolic network directly on the manifold for spatio-temporal graphs to perform network operations without resorting to the tangent space and without normalization.
- We introduce an attention module to dynamically infer the relationship between graph nodes on the hyperbolic manifold without the need for a graph static adjacency matrix as was the case for other methods.
- We evaluate the performance on traffic forecasting datasets and show that our proposed method achieves better

Manuscript received 12 December 2023; revised 19 March 2024; accepted 9 May 2024. Date of publication 15 May 2024; date of current version 23 May 2024. This work was supported in part by the Research Council of Finland (former Academy of Finland) Academy Professor project EmotionAI under Grant 336116 and Grant 345122 and in part by the University of Oulu and Research Council of Finland Profi 7 under Grant 352788. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Sundeep Prabhakar Chepuri. *(Corresponding author: Guoying Zhao.)*

The authors are with the Center for Machine Vision and Signal Analysis, University of Oulu, 90570 Oulu, Finland (e-mail: abdelrahman.mostafa@oulu.fi; guoying.zhao@oulu.fi).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LSP.2024.3401619, provided by the authors.

Digital Object Identifier 10.1109/LSP.2024.3401619

performance with less number of parameters compared to other Euclidean methods using small models.

II. Proposed Method

A. Graph Notations

A static graph is a set of nodes with features and a set of edges i.e., $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ where $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ is the set of n graph nodes and $\mathcal{E}$ is the set of graph edges that connect the graph nodes. The edge set $\mathcal{E}$ can be encoded in an adjacency matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ where $\mathbf{A}_{i,j} \in [0, 1)$ can represent the weight of the edge between graph nodes v_i and v_j. Each node v_i has a feature vector x_i of dimension d and $\mathbf{X} = \{x_1, x_2, \ldots, x_n\}$ is the set of feature vectors of all graph nodes. A spatio-temporal graph evolves across the time domain and can be thought of as a set of static graphs i.e., $\mathcal{S} = \{G_1, G_2, \ldots, G_F\}$ where F is the number of frames or observations for this graph with $\mathbf{X}_t$ as the features of the graph nodes at time step t.

B. Lorentz Transformations

Lorentz transformations are manifold-preserving transformations which can be used to directly transform graph features $\mathbf{X}_t$ on the hyperbolic manifold [15]. A Lorentz transformation matrix Λ is a linear transformation matrix for the hyperbolic space (refer to [25] for an introduction to hyperbolic geometry). The matrix Λ must then satisfy the following constraint:

$$\Lambda^T g_{\mathcal{L}} \Lambda = g_{\mathcal{L}} \tag{1}$$

where $\Lambda \in \mathbb{R}^{(d+1)\times(d+1)}$ and T represents the transpose operation of the matrix. Λ belongs to the Lorentz group and is orthogonal with respect to the Minkowski metric $g_{\mathcal{L}}$ where $g_{\mathcal{L}} = diag(-1, 1, \ldots, 1)$ is a diagonal matrix that represents the Riemannian metric for the Lorentz model of the hyperbolic manifold [26]. Note that for the case of the Euclidean space with the metric as the identity matrix, the transformation matrix is orthogonal with respect to the Euclidean metric in a similar way. The Lorentz transformation can be decomposed into a spatial rotation and a boost operations [15] and [13] by polar decomposition. The spatial rotation operation rotates the spatial coordinates whereas the boost operation moves a point along the time coordinate without rotating the spatial coordinates. The spatial rotation matrix is given by:

$$\mathbf{P} = \begin{bmatrix} 1 & 0 \\ 0 & \mathbf{O} \end{bmatrix}_{(d+1)\times(d+1)} \tag{2}$$

where $\mathbf{O}$ belongs to the special orthogonal group $SO(d)$ i.e., $\mathbf{O^T O = I}$. It can easily verified that $\mathbf{P}$ indeed satisfies (1). $\mathbf{O}$ can be learnt by restricting the learnt kernel to be orthogonal. The boost matrix is:

$$\mathbf{L} = \begin{bmatrix} \cosh\omega & (\sinh\omega) n_d^T \\ (\sinh\omega) n_d & \mathbf{I} - (1 - \cosh\omega) n_d \otimes n_d \end{bmatrix}_{(d+1)\times(d+1)} \tag{3}$$

where n_d is the hyperbolic rotation axis and ω is the hyperbolic rotation parameter (similar to the case using a circular rotation axis and a circular rotation parameter for circular\regular rotation). $\otimes$ represents the outer product operation. It can be shown that $\mathbf{L}$ satisfies (1) as well. Since both $\mathbf{P}$ and $\mathbf{L}$ satisfy (1), their product $\mathbf{PL}$ satisfies the Equation as $(\mathbf{PL})^T g_{\mathcal{L}} (\mathbf{PL}) = \mathbf{L}^T(\mathbf{P}^T g_{\mathcal{L}} \mathbf{P})\mathbf{L} = \mathbf{L}^T g_{\mathcal{L}} \mathbf{L} = g_{\mathcal{L}}$.

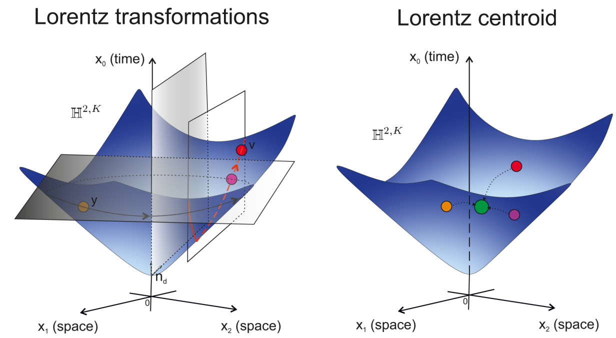

Fig. 1. Feature transformation and aggregation steps in the proposed method.

C. Learning Features Across Spatial Domain in LSTGCN

The Lorentz transformation matrix $\mathbf{PL}$ can be used for the feature transformation step for the spatial module in LSTGCN. This can be represented as:

$$\mathbf{Y}_t^{h,l} = \mathbf{X}_t^{h,l} \mathbf{P}^l \mathbf{L}^l \tag{4}$$

where h stands for hyperbolic features at a frame t for a layer l in the network. To get the initial hyperbolic features for the first layer, the exponential map [1] is used to map the features from the tangent space at the origin to the hyperboloid.

For feature aggregation, the Lorentz centroid [27], [28] which minimizes the squared Lorentzian distance can be used:

$$x_{t,i}^{h,l+1} = \sqrt{K} \frac{\sum_{j\in NS(i)} w_{i,j} y_{t,j}^{h,l}}{|\|\sum_{j\in NS(i)} w_{i,j} y_{t,j}^{h,l}\|_{\mathcal{L}}|} \tag{5}$$

where ${}^{-1}/_K$ is the constant negative curvature for the hyperboloid ($K > 0$), $NS(i)$ is the neighbor set for node v_i which includes the node itself and $w_{i,j}$ is the weight of the edge between nodes v_i and v_j, for example, the element $\mathbf{A}_{i,j}$ in the adjacency matrix $\mathbf{A}$. Fig. 1 shows a visualization for feature transformation and aggregation steps using Lorentz transformations (left) and Lorentz centroid (right), respectively.

D. Attention Module for Learning Similarities in LSTGCN

The weight of the edge $w_{i,j}$ between nodes v_i and v_j can be learnt using attention values instead of using fixed weights from the graph adjacency matrix $\mathbf{A}$. A Lorentz transformation matrix Λ can be used as the weights for forming the keys and queries. The keys are then $\mathbf{K}_t^{h,l} = \mathbf{X}_t^{h,l}\Lambda_k^l$ and the queries are $\mathbf{Q}_t^{h,l} = \mathbf{X}_t^{h,l}\Lambda_q^l$ where Λ_k^l and Λ_q^l are the Lorentz transformation weights for the keys and the queries, respectively. To determine the similarity between two nodes on the Lorentz manifold, the squared Lorentzian distance can be used. The squared Lorentzian distance between the hyperbolic embeddings $x_{t,i}^{h,l}$ and $x_{t,j}^{h,l}$ of the graph nodes v_i and v_j at time t for layer l has the form:

$$d_{\mathcal{L}}^2(x_{t,i}^{h,l}, x_{t,j}^{h,l}) = -2\langle K + \langle x_{t,i}^{h,l}, x_{t,j}^{h,l}\rangle_{\mathcal{L}}\rangle \tag{6}$$

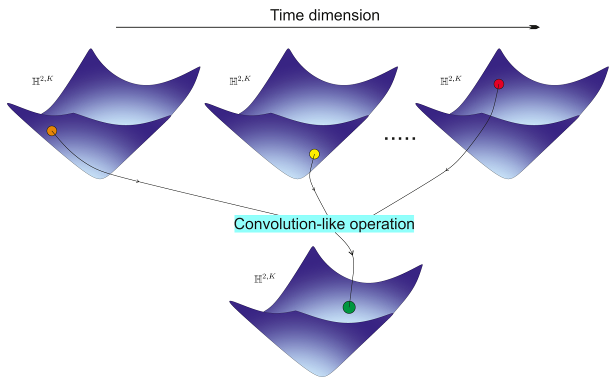

Fig. 2. Propagating features between graph nodes across the time domain.

where $\langle ., . \rangle_{\mathcal{L}} : \mathbb{R}^{d+1} \times \mathbb{R}^{d+1} \to \mathbb{R}$ is the Lorentz-Minkowski inner product where $\langle x, y \rangle_{\mathcal{L}} := \sum_{i=1}^{d} x_i y_i - x_0 y_0 = x^T g_{\mathcal{L}} y$. The edge weight $w_{i,j}^{t,l}$ between a hyperbolic query $q_{t,i}^{h,l}$ and a hyperbolic key $k_{t,j}^{h,l}$ for graph nodes v_i and v_j can then be computed as:

$$w_{i,j}^{t,l} = \frac{\exp(-d_{\mathcal{L}}^2(q_{t,i}^{h,l}, k_{t,j}^{h,l})/\sqrt{n})}{\sum_{m=1}^{n} \exp(-d_{\mathcal{L}}^2(q_{t,i}^{h,l}, k_{t,m}^{h,l})/\sqrt{n})} \tag{7}$$

These calculated weights give attention to nodes depending on the similarity between nodes.

E. Learning Features Across Temporal Domain in LSTGCN

In Euclidean ST-GCNs, each node can be connected to the corresponding node in the previous frame (except for the first frame) and to the next frame (except for the last frame) to propagate features between nodes through the temporal domain. For a kernel $\Gamma^{d_{out} \times r \times d_{in}}$ where d_{out} is the dimension of the output feature, d_{in} is the dimension of the input feature and r is the length of the kernel. The output Euclidean feature at index j for layer l at a given frame t can be expressed as:

$$x_{t,j}^{e,l} = \sum_{i=1}^{r} \Gamma^{j,i} x_{t-(r+1)/2+i}^{e,l} \tag{8}$$

Similarly, we need to design a similar operation for the hyperbolic space which is manifold-preserving and having the same number of parameters i.e., using a kernel $\Gamma^{d_{out} \times r \times d_{in}}$ for it to be cheap as the case for the Euclidean space. Fig. 2 shows a visualization of the tangent space-free convolution-like operation across the time domain that is needed to propagate features between different time steps. The convolution-like operation must satisfy the manifold-preserving condition i.e., ${\Lambda_t}^T g_{\mathcal{L}} \Lambda_t = g_{\mathcal{L}}$ for each graph t in the sequence. Since the convolution operation for the Euclidean space in (8) is an inner product i.e., scaling each feature index entry by the corresponding kernel index entry then followed by summation, we propose an operation using hyperbolic rotations which can be thought of as a scaling of the corresponding spatial dimension. The hyperbolic rotation using a hyperbolic rotation parameter ω can be expressed as:

$$\mathbf{R} = \begin{bmatrix} \cosh \omega & \sinh \omega \\ \sinh \omega & \cosh \omega \end{bmatrix} \tag{9}$$

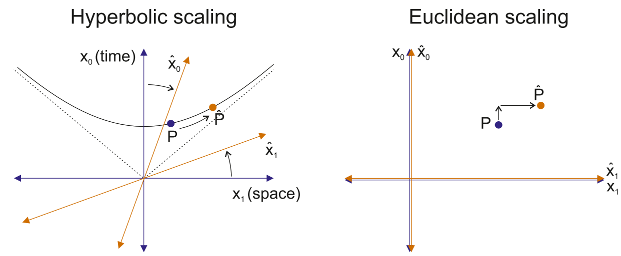

Fig. 3. Hyperbolic rotation for hyperbolic scaling vs scalars for scaling.

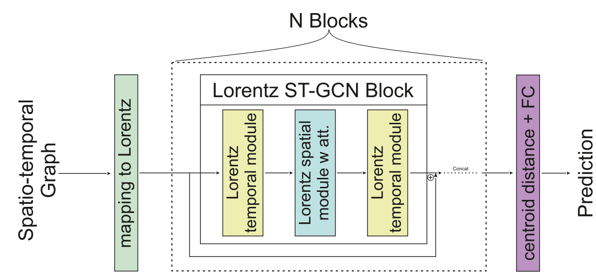

Fig. 4. Tangent space-free network for traffic forecasting.

Fig. 3 shows a visualization of the scaling using hyperbolic rotations for hyperbolic spaces and the scaling using scalars for Euclidean spaces. The hyperbolic rotation operation is manifold-preserving and the summation operation in the Euclidean space case can be replaced by the Lorentz centroid (5) which is manifold-preserving for the hyperbolic space. It is also worth mentioning that the general hyperbolic rotation using (3) can be used also with the same number of parameters using a kernel $\Gamma^{d_{out} \times r \times d_{in}}$ i.e., using the same kernel size.

F. LSTGCN Full Network Architecture

The design of the tangent space-free Lorentz ST-GCN for the traffic forecasting regression task is shown in Fig. 4. A residual skip connection is used which is implemented using the Lorentz centroid in (5). To obtain the final prediction, the distance between the node feature and a set of trainable centroids on the manifold is calculated which is then fed to a fully connected layer to predict the output for the traffic forecasting regression tasks.

III. Experimental Results

A. Datasets

Traffic data can be represented as spatio-temporal graph to predict the traffic flow, for instance. We use four traffic datasets from California, namely, PEMS03, PEMS04, PEMS07 and PEMS08 [22]. The traffic data collected from traffic sensors is aggregated into 5-minute intervals. The task is to predict the traffic flow in the next hour (12 time steps) given the traffic flow data of the last hour. For performance evaluation, the mean absolute error (MAE) and the root mean squared error (RMSE) are used. The same data split is used as in previous works to divide the data into training set, validation set and test set as 6:2:2.

TABLE I
PERFORMANCE COMPARISON BETWEEN DIFFERENT METHODS. H STANDS FOR HYPERBOLIC

	Dataset		PEMS03		PEMS04		PEMS07		PEMS08		Parameters	
	Method	Dim	MAE (↓)	RMSE (↓)	MAE (↓)	RMSE (↓)	MAE (↓)	RMSE (↓)	MAE (↓)	RMSE (↓)	Count	Ratio
Euclidean	TGCN [29]	32	45.96	66.51	48.00	71.60	49.47	72.52	59.74	85.77	6.83K	1.8x
	A3TGCN [30]	32	48.88	69.53	50.93	74.81	53.86	77.58	62.03	87.88	6.84K	1.8x
	AGCRN [18]	4	**18.91**	32.85	30.02	52.34	33.53	55.39	65.22	107.93	4.31K	1.1x
	STGCN [5]	8	19.71	32.20	25.16	38.96	29.00	44.28	20.39	31.61	5.64K	1.5x
	ASTGCN [23]	4	19.54	33.84	**23.95**	**37.41**	30.76	47.15	20.80	33.12	1.56M	403.1x
	DSTAGNN [31]	32	32.49	49.62	41.22	61.11	52.66	74.58	45.08	62.00	3.40M	878.6x
H	LSTGCN (Ours)	4	19.38	**31.92**	25.10	39.61	**28.04**	**43.80**	**19.90**	**31.59**	**3.87K**	**1.0x**

B. Performance Evaluation

For performance comparison, the following baselines which use the Euclidean space for features embedding are used: 1) TGCN [29] is a recurrent network that used GCN to model the spatial relationships between nodes and a gated recurrent unit to capture the temporal relationships; 2) A3TGCN [30] added attention parameters to TGCN to capture the relevant features across the time domain; 3) AGCRN [18] used adaptive modules to enhance the performance of the recurrent GCN; 4) STGCN [5] used GCN to model the spatial dependence and a temporal 1D convolution to propagate feature across the time domain; 5) ASTGCN [23] introduced attention modules to the design of STGCN; 6) DSTAGNN [31] used multi-scale gated convolution and pre-processed graphs for capturing spatio-temporal dynamics to improve the performance. We scaled down the baseline methods to obtain light-weight compact models with the best possible performance.

Table I shows the performance comparison between different methods using MAE and RMSE metrics. MAE is a linear operator (L1 norm) which gives equal weights to each error whereas RMSE is a quadratic operator (L2 norm) which penalizes big errors. The Dim column in the table represents the dimension for the latent features used in the model. The Count column shows the number of parameters in each model and the Ratio column shows the count ratio with respect to the parameter count in LSTGCN. LSTGCN achieved the best performance compared to other methods for most datasets using the least number of parameters. ASTGCN achieved a better performance on the PEMS04 dataset but ASTGCN has far more number of parameters (1.56 M vs 3.87 K) than LSTGCN or about 403.1 times the number of parameters in our method. The results show the effectiveness of using hyperbolic networks for embedding spatio-temporal graphs using light-weight compact models having less number of parameters. In addition, LSTGCN uses attention to determine similarities between nodes without using the graph adjacency matrix as the case for other methods.

C. Ablation Study

Table II shows an ablation study to compare the performance after removing one of the components from LSTGCN. Removing temporal or spatial modules decreases the model performance as these modules propagate features through the temporal and spatial domains, respectively. The attention module helps the network to focus on relevant information in the input sequence. Adding a path as a residual skip connection helps in back-propagation and updating the model weights. Fig. 5 shows a visualization for the error of different time steps on the PEMS03 dataset. Adding more modules to the hyperbolic network that are manifold-preserving can potentially improve the performance of hyperbolic networks. More analysis can be found in the supplementary materials and the code is available at: https://github.com/leoamb/LSTGCN.

TABLE II
ABLATION STUDY TO SHOW THE EFFECT OF DIFFERENT MODULES

Dataset	PEMS03		PEMS04		PEMS07		PEMS08	
Method	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
LSTGCN	**19.38**	**31.92**	**25.10**	**39.61**	**28.04**	**43.80**	**19.90**	**31.59**
W/o tem.	19.88	32.13	25.42	40.20	28.47	44.11	20.01	31.66
W/o att.	20.71	33.80	26.42	41.14	30.19	38.46	21.30	33.35
W/o spa.	20.92	34.24	26.86	41.96	30.35	46.41	21.83	34.14
W/o skip	24.12	38.46	128.39	161.31	157.15	187.37	23.43	35.57

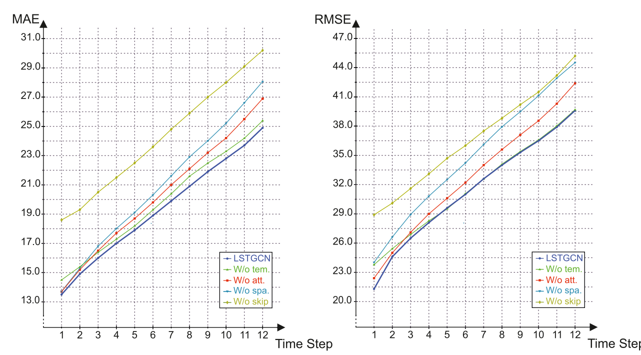

Fig. 5. MAE and RMSE for different time steps on the PEMS03 dataset.

IV. CONCLUSION

We introduced a tangent space-free Lorentz Spatial Temporal Graph Convolution Network to embed spatio-temporal graphs. The performance was evaluated on the traffic forecasting task and very good results were achieved using compact hyperbolic models with small number of model parameters which can make it suitable for mobile applications. The promising results show the superiority and the advantage of using the hyperbolic space over the traditional Euclidean space. This work can be extended by adding more manifold-preserving operations that can potentially increase the effectiveness of hyperbolic networks.

REFERENCES

[1] I. Chami, Z. Ying, C. Ré, and J. Leskovec, "Hyperbolic graph convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 4868–4879.

[2] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 3482–3489.

[3] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2017. [Online]. Available: https://openreview.net/forum?id=SJU4ayYgl

[4] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3844–3852.

[5] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 3634–3640.

[6] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Two-stream adaptive graph convolutional networks for skeleton-based action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 12018–12027.

[7] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Skeleton-based action recognition with multi-stream adaptive graph convolutional networks," *IEEE Trans. Image Process.*, vol. 29, pp. 9532–9545, 2020.

[8] W. Peng, T. Varanka, A. Mostafa, H. Shi, and G. Zhao, "Hyperbolic deep neural networks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 12, pp. 10023–10044, Dec. 2022.

[9] A. Mostafa, W. Peng, and G. Zhao, "Hyperbolic spatial temporal graph convolutional networks," in *Proc. IEEE Int. Conf. Image Process.*, 2022, pp. 3301–3305.

[10] Q. Liu, M. Nickel, and D. Kiela, "Hyperbolic graph neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8230–8241.

[11] Y. Zhang, X. Wang, C. Shi, N. Liu, and G. Song, "Lorentzian graph convolutional networks," in *Proc. Web Conf.*, 2021, pp. 1249–1261.

[12] C. Battiloro, Z. Wang, H. Riess, P. Di Lorenzo, and A. Ribeiro, "Tangent bundle filters and neural networks: From manifolds to cellular sheaves and back," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2023, pp. 1–5.

[13] W. Chen et al., "Fully hyperbolic neural networks," in *Proc. 60th Ann. Meeting Assoc. Comput. Linguistics*, 2022, pp. 5672–5686, doi: 10.18653/v1/2022.acl-long.389.

[14] J. Dai, Y. Wu, Z. Gao, and Y. Jia, "A hyperbolic-to-hyperbolic graph convolutional network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 154–163.

[15] A. Mostafa, W. Peng, and G. Zhao, "SRBGCN: Tangent space-Free Lorentz transformations for graph feature learning," in *Proc. Brit. Mach. Vis. Assoc.*, 2023.

[16] J. Zhang, Y. Zheng, D. Qi, R. Li, and X. Yi, "DNN-based prediction model for spatio-temporal data," in *Proc. 24th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, 2016, pp. 1–4.

[17] Y. Li and C. Shahabi, "A brief overview of machine learning methods for short-term traffic forecasting and future directions," *Sigspatial Special*, vol. 10, no. 1, pp. 3–9, 2018.

[18] L. Bai, L. Yao, C. Li, X. Wang, and C. Wang, "Adaptive graph convolutional recurrent network for traffic forecasting," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 17804–17815.

[19] C. Zheng, X. Fan, C. Wang, and J. Qi, "GMAN: A graph multi-attention network for traffic prediction," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 1234–1241.

[20] S. Fang, X. Pan, S. Xiang, and C. Pan, "Meta-MSNet: Meta-learning based multi-source data fusion for traffic flow prediction," *IEEE Signal Process. Lett.*, vol. 28, pp. 6–10, 2021.

[21] J. Sun, X. Xie, D. Teng, and X. Ju, "Traffic flow prediction via spatial-temporal geometric graph convolutional network," in *Proc. IEEE 5th Adv. Inf. Manage., Communicates, Electron. Automat. Control Conf.*, 2022, pp. 1106–1110.

[22] C. Song, Y. Lin, S. Guo, and H. Wan, "Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 914–921.

[23] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 922–929.

[24] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. Int. Conf. Learn. Representations*, 2018.

[25] R. Benedetti and C. Petronio, *Lectures on Hyperbolic Geometry*. Berlin, Germany: Springer, 1992.

[26] H. Minkowski, *Raum Und Zeit*. Berlin, Germany: Springer, 1988.

[27] J. G. Ratcliffe, S. Axler, and K. Ribet, *Foundations of Hyperbolic Manifolds*, vol. 149. Berlin, Germany: Springer, 1994.

[28] M. Law, R. Liao, J. Snell, and R. Zemel, "Lorentzian distance learning for hyperbolic representations," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 3672–3681.

[29] L. Zhao et al., "T-GCN: A temporal graph convolutional network for traffic prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 9, pp. 3848–3858, Sep. 2020.

[30] J. Bai et al., "A3T-GCN: Attention temporal graph convolutional network for traffic forecasting," *ISPRS Int. J. Geo- Inf.*, vol. 10, no. 7, 2021, Art. no. 485.

[31] S. Lan, Y. Ma, W. Huang, W. Wang, H. Yang, and P. Li, "DSTAGNN: Dynamic spatial-temporal aware graph neural network for traffic flow forecasting," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 11906–11917.