STFGNN

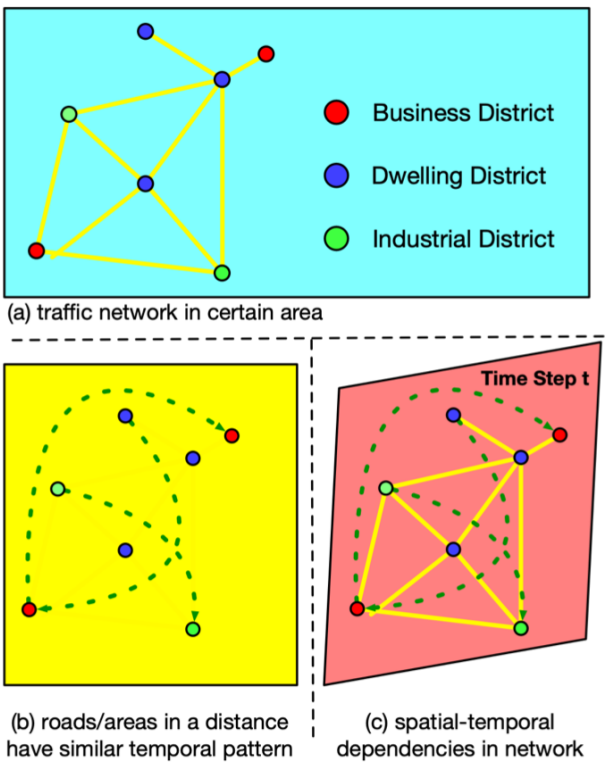
Spatial-Temporal Fusion Graph Neural Networks for Traffic Flow Forecasting AAAI2021

研究背景：

过去基于图神经网络的时空序列预测模型缺少空间相关性的考虑：忽略了两个节点即使是地理上不相邻，在空间上也有强相关性。比如在早晚高峰时间，科技园和大厦附近的拥堵规律相似。为了能够让模型学习到时空间信息之外的特征做了如下尝试：

通过改变邻接矩阵的模型：STSGCN与Graph WaveNet两个模型提出了改变本来的图邻接矩阵来刻画空间相关度，但这些自适应的邻接矩阵无法描述复杂的时序特征（两个节点之间的关系有很多种：周期变化相似，变化相反，A堵完B会接着堵等等）。其中STSGCN使用的是Masked Matrix，而Graph WaveNet自己训练邻接矩阵。

应用注意力的模型：而STTNs 与 STGNN 应用了时间上的self-attention机制应用到捕捉时空序列特征的模型。注意力机制虽然能捕捉到预训练出的时空序列矩阵捕捉不到的序列信息，但很容易因为时间信息的动态变化和信息噪音，在很长的序列预测中过拟合。



而在捕获长时间的时间序列信息上，过去的模型都不能够有效率的捕捉就近时间点和过去时间点之间的关系。RNN和LSTM的训练时间都过长，而且会有梯度爆炸/消失的问题。使用Transformer训练也不快。基于

CNN的方式需要叠加很多层才能够捕获长时间序列的信息。使用dilation rate过大的空洞卷积又会丢失临近时间点的信息。

本文提出两个创新点：DTW和Gated空洞卷积CNN。前者用于捕捉空间临近关系之外的信息，后者用于捕捉长时间的规律特征。

模型针对以上问题提出了基于Dynamic Time Warping算法构建时间序列相似度矩阵，用于捕捉地理上不相关的节点相似信息。并将模型输入并行输入到两个模块STFGN模块与Gated CNN模块，用于捕捉地理上的特征。

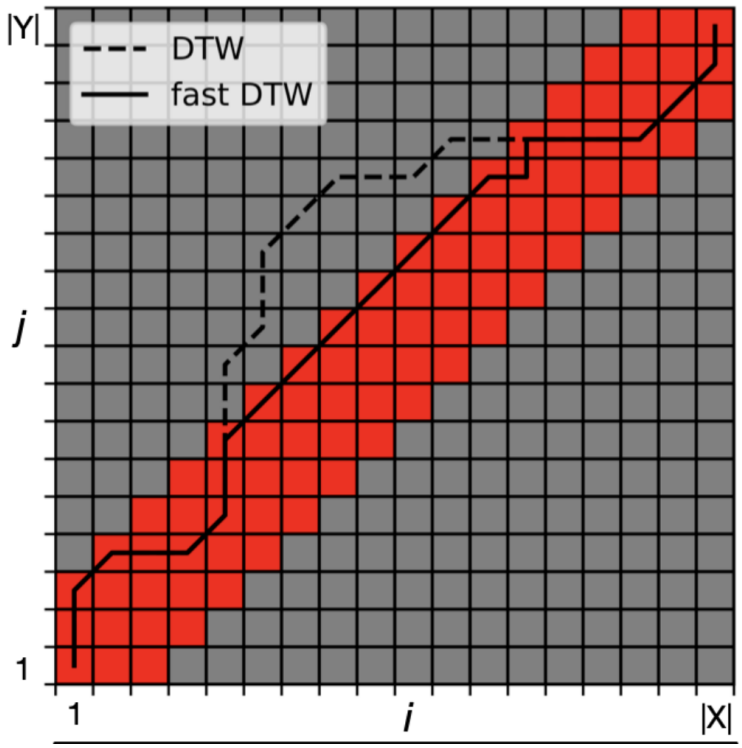
DTW算法：计算时间序列X与时间序列Y的相似度。

首先构建时间序列x与时间序列y之间差值矩阵M，其中

$$M_{i,j} = |x_i - y_j|$$

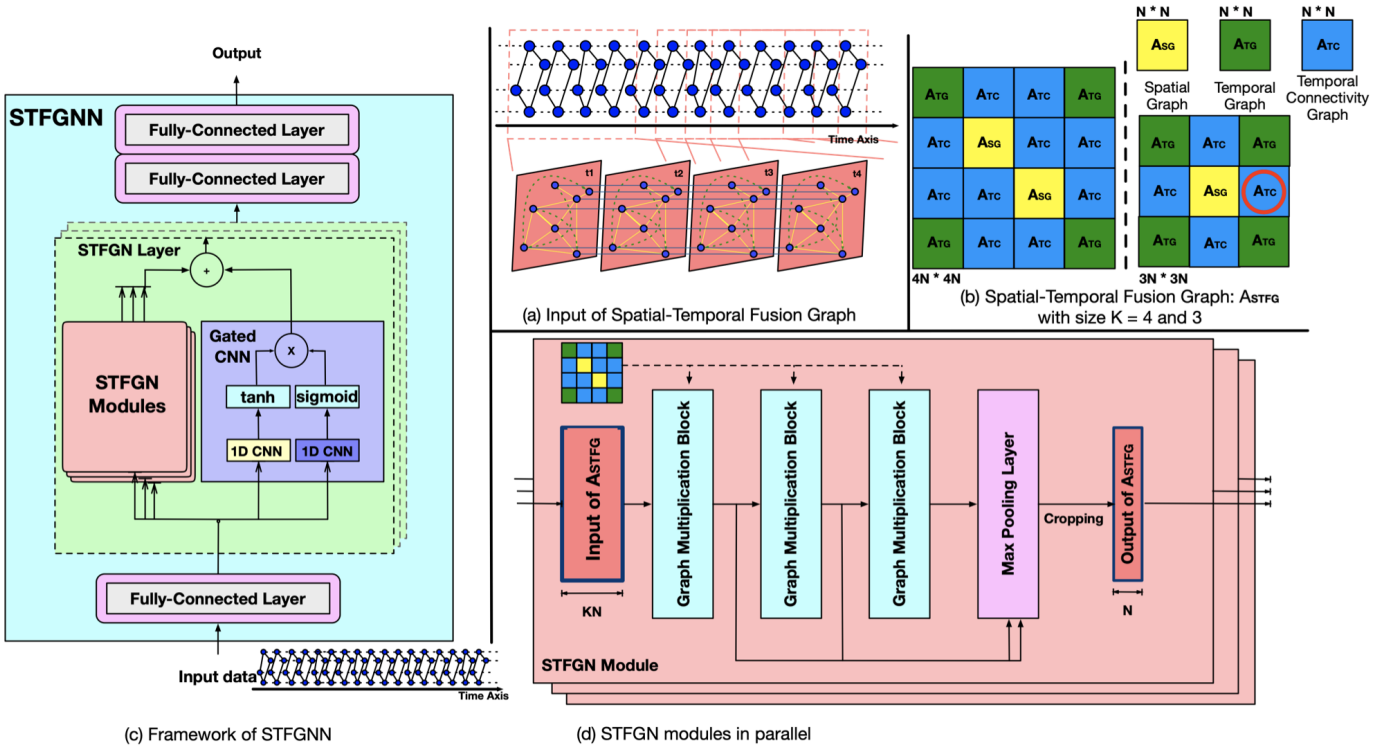
然后基于动态规划算法构建相似度矩阵F，其中转移方程如下转移方程如下

$$F_{i,j} = M_{i,j} + min(F_{i-1,j}, F_{i,j-1}, F_{i-1,j-1})$$



如图为最短路径的动态规划图示意：Fast DTW把搜寻的路径控制在了红色的区域内，加快了训练速度。

模型



模型一些要注意的：

1）构造时间矩阵ATG时，对每个节点找出与其序列最相似的k个节点，并在这些节点间构建边。k为经验值，其大小为邻接矩阵A的平均度数。这样是为了使邻接矩阵ASG与时间矩阵ATG的稀疏程度相同。

Algorithm 1: Temporal Graph Generation

Input: N time series from $\mathcal{V}(|\mathcal{V}| = N)$

W Initialization, reset to zero matrix TDL: Temporal Distance Calculation defined in Alg 2

for $i = 1, 2, \cdots, N$ **do**

  for $j = 1, 2, \cdots, N$ **do**

    $dist_{i,j}$ = TDL(V_i, V_j) (Alg. 2)

  end

  Sort smallest $k(k \leq N)$ element and their index $\mathbf{j} = \{j_1, j_2, \cdots, j_k\}$ s.t. $dist_{i,j_1} \leq dist_{i,j_2} \leq dist_{i,j_k}$ **if** $\tilde{j} \in \mathbf{j}$ **then** $W_{i,\tilde{j}} = W_{\tilde{j},i} = 1$;

end

return Weighted Matrix W of Temporal Graph $\mathcal{G}$.

2）蓝色的矩阵ATC为单位矩阵。

```
for i in range(N):
    for k in range(steps - 1):
        adj[k * N + i, (k + 1) * N + i] = 1
        adj[(k + 1) * N + i, k * N + i] = 1
....
```

3）输入的矩阵维度：(B, T, N, C)， 大的邻接矩阵维度为（4N，4N）。相乘时先截取4个时间截点(t1,t2,t3,t4)，变换维度成(4N, B, C)，然后与矩阵相乘。然后再截取下四个时间点(t2, t3, t4, t5)，把输入矩阵再变成（4N, B, C)维与矩阵相乘。

```
Python
1     for i in range(T - 3):
2         # shape is (B, 4, N, C)
```

```
        t = mx.sym.slice(data, begin=(None, i, None, None),
                         end=(None, i + 4, None, None))
        # shape is (B, 4N, C)
        t = mx.sym.reshape(t, (-1, 4 * num_of_vertices, num_of_features))
        # shape is (4N, B, C)
        t = mx.sym.transpose(t, (1, 0, 2))
        # shape is (N, B, C')
        t = stsgcm(
            t, adj, filters, num_of_features, num_of_vertices,
            activation=activation,
            prefix="{}_stsgcm_{}".format(prefix, i)
        )
        # shape is (B, N, C')
        t = mx.sym.swapaxes(t, 0, 1)
        # shape is (B, 1, N, C')
        need_concat.append(mx.sym.expand_dims(t, axis=1))
    # shape is (B, T-3, N, C')
    #return mx.sym.concat(*need_concat, dim=1)
    need_concat_ = mx.sym.concat(*need_concat, dim=1)
    layer_out = need_concat_ + data_res
    return layer_out
```

实验结果

S代表空间相邻矩阵在四个角，没有S代表空间邻接矩阵全部换成了刻画序列相似度的矩阵。

T3，T4代表大的融合矩阵是3*3 还是 4*4

Tsp5与Tsp1代表k的取值控制刻画序列相似度的矩阵有5%还是1%的「1」

theta代表着时间gated cnn有没有加进去。

Datasets	Metric	FC-LSTM	DCRNN	STGCN	ASTGCN(r)	Graph WaveNet	STSGCN	STFGNN
PEMS03	MAE	21.33 ± 0.24	18.18 ± 0.15	17.49 ± 0.46	17.69 ± 1.43	*19.85 ± 0.03*	17.48 ± 0.15	**16.77 ± 0.09**
	MAPE(%)	23.33 ± 4.23	18.91 ± 0.82	17.15 ± 0.45	19.40 ± 2.24	*19.31 ± 0.49*	16.78 ± 0.20	**16.30± 0.09**
	RMSE	35.11 ± 0.50	30.31 ± 0.25	30.12 ± 0.70	29.66 ± 1.68	*32.94 ± 0.18*	29.21 ± 0.56	**28.34± 0.46**
PEMS04	MAE	27.14 ± 0.20	24.70 ± 0.22	22.70 ± 0.64	22.93 ± 1.29	*25.45 ± 0.03*	21.19 ± 0.10	**19.83± 0.06**
	MAPE(%)	18.20 ± 0.40	17.12 ± 0.37	14.59 ± 0.21	16.56 ± 1.36	*17.29 ± 0.24*	13.90 ± 0.05	**13.02± 0.05**
	RMSE	41.59 ± 0.21	38.12 ± 0.26	35.55 ± 0.75	35.22 ± 1.90	*39.70 ± 0.04*	33.65 ± 0.20	**31.88± 0.14**
PEMS07	MAE	29.98 ± 0.42	25.30 ± 0.52	25.38 ± 0.49	28.05 ± 2.34	*26.85 ± 0.05*	24.26 ± 0.14	**22.07± 0.11**
	MAPE(%)	13.20 ± 0.53	11.66 ± 0.33	11.08 ± 0.18	13.92 ± 1.65	*12.12 ± 0.41*	10.21 ± 1.65	**9.21± 0.07**
	RMSE	45.94 ± 0.57	38.58 ± 0.70	38.78 ± 0.58	42.57 ± 3.31	*42.78 ± 0.07*	39.03 ± 0.27	**35.80± 0.18**
PEMS08	MAE	22.20 ± 0.18	17.86 ± 0.03	18.02 ± 0.14	18.61 ± 0.40	*19.13 ± 0.08*	17.13 ± 0.09	**16.64± 0.09**
	MAPE(%)	14.20 ± 0.59	11.45 ± 0.03	11.40 ± 0.10	13.08 ± 1.00	*12.68 ± 0.57*	10.96 ± 0.07	**10.60± 0.06**
	RMSE	34.06 ± 0.32	27.83 ± 0.05	27.83 ± 0.20	28.16 ± 0.48	*31.05 ± 0.07*	26.80 ± 0.18	**26.22± 0.15**

Dataset	Model Elements	MAE	MAPE%	RMSE
PEMS04	STSGCN	21.19	13.90	33.65
	$[ST_3, T_{sp5}]$	20.74	13.77	33.44
	$[ST_3, T_{sp1}]$	20.09	13.24	32.44
	$[ST_4, T_{sp1}]$	19.92	13.03	31.93
	$[T_4, T_{sp1}, \Theta]$	20.02	13.17	31.98
	$[T_4, T_{sp5}, \Theta]$	19.91	13.11	32.19
	$[ST_4, T_{sp1}, \Theta]$	**19.83**	**13.02**	**31.88**
PEMS08	STSGCN	17.13	10.96	26.80
	$[ST_3, T_{sp5}]$	19.47	12.27	29.59
	$[ST_3, T_{sp1}]$	16.84	10.80	26.58
	$[ST_4, T_{sp1}]$	16.70	10.63	26.24
	$[T_4, T_{sp1}, \Theta]$	18.23	11.52	29.05
	$[T_4, T_{sp5}, \Theta]$	**16.02**	**10.07**	**25.39**
	$[ST_4, T_{sp1}, \Theta]$	16.64	10.60	26.22

https://github.com/MengzhangLI/STFGNN MXNet