Multi-Model Traffic Flow Prediction

25AI60R28 – Jaykumar Lokhande
25AI60R23 – Aishik Das
25AI60R27 – Karan Kavatra
24IM92R03 – Ravi Raj Anand

Under Guidance of Prof. Mahesh Mohan

Table of contents

1

Motivation & Literature Review

Problem Statement

Predict future traffic flow from historical sensor data

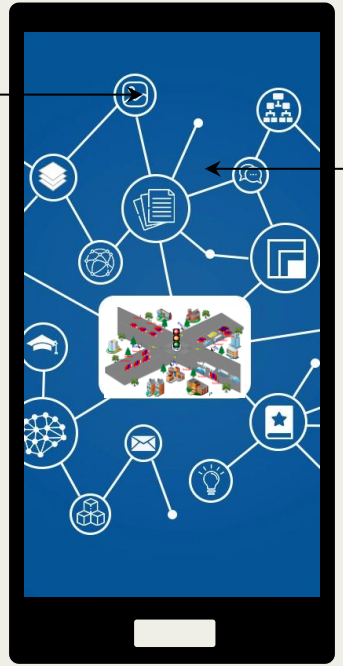
Complexity:
1. Spatial dependencies (sensor relationships)
2. Temporal patterns (time series)
3. Multi-feature data (flow, occupancy, speed) or (inflow, outflow)

Applications:
Traffic management, route optimization, congestion prediction

Why the need of GCN???

Why the need though?

Traffic jams waste time, fuel, and money. Can we forecast them before they form?



Why not Classical ML ?

- Treats each sensor as a single feature vector.
- Incapable of realizing graphical and temporal correlation.
- Hence poor generalization.

Motivation

Baseline Methods		VAR	SVR
Datasets	Evaluation Metrics		
PeMS04	RMSE	36.66	44.59
	MAE	23.75	28.66
	MAPE (%)	18.09	19.15
PeMS08	RMSE	33.83	36.15
	MAE	22.32	23.25
	MAPE (%)	14.47	14.71

[1] Table 1 : Traditional Models

Literature Review

Statistical Approaches (1980s-2000s)

ARIMA Models (Williams & Hoel, 2003):

- Auto Regressive, Integrated, Moving Average
- **Advantages**: Simple, interpretable, good for short-term
- **Limitations**: Linear assumptions, can't capture spatial dependencies
- **Performance**: Baseline ~15-20% MAE on traffic datasets

Kalman Filtering (Okutani & Stephanedes, 1984):

- Dynamic state estimation with uncertainty
- **Advantages**: Real-time updates, handles noise
- **Limitations**: Linear state transitions, limited scalability
- **Application**: Traffic state estimation, sensor fusion

Literature Review

Machine Learning Approaches (2000s-2010s)

Support Vector Machines (SVM) (Wu et al., 2004):

- Non-linear regression with kernel methods
- **Advantages**: Better non-linear modelling than ARIMA
- **Limitations**: Limited scalability, no temporal structure
- **Performance**: ~10-15% improvement over ARIMA

Random Forests (Breiman, 2001):

- Ensemble of decision trees
- **Advantages**: Handles non-linearity, feature importance
- **Limitations**: No explicit temporal/spatial modelling
- **Application**: Multi-feature traffic prediction

Key Gap: None of these methods capture both spatial and temporal dependencies simultaneously

Literature Review

Early Deep Learning (2010-2015)

Convolutional Neural Networks (CNNs) (Zhang et al., 2017):

- Applied 1D CNNs to temporal traffic patterns
- **Advantage**: Captures local temporal patterns
- **Limitation**: Treats sensors independently (no spatial modelling)
- **Performance**: ~8-12% improvement over traditional methods

Recurrent Neural Networks (RNNs):

- **LSTM** (Hochreiter & Schmidhuber, 1997): Long-term memory
- **GRU** (Cho et al., 2014): Simplified LSTM variant
- **Application**: Ma et al. (2015) applied LSTM to traffic speed
- **Advantage**: Handles temporal sequences effectively
- **Limitation**: No spatial relationships, sequential processing

Literature Review

Graph Neural Networks Revolution (2017-2018)

Graph Convolutional Networks (GCN) (Kipf & Welling, 2017):

- **Key Innovation**: Spectral graph convolution using Chebyshev approximation
- **Impact**: Foundation for graph-based traffic prediction
- **Efficiency**: O(K × |E|) complexity, where K=polynomial order

Chebyshev Graph Convolution (Defferrard et al., 2016):

- Fast localized spectral filtering
- **Advantage**: Computationally efficient, localized filters
- **Used in**: STGCN for spatial modelling

Spatio-Temporal Integration (2018+)

- **STGCN** (Yu et al., 2018): First unified spatio-temporal framework
- **GAT** (Veličković et al., 2018): Attention-based graph learning

Papers

LSTM for Traffic Prediction (2015) — Ma et al.

Node-wise LSTM/GRU for time-series forecasting

STGCN: Spatial-Temporal Graph Convolutional Networks (2017)

ChebConv + temporal conv + ST-Conv blocks

STGAT: Spatio-Temporal Graph Attention Network (2018)

GATConv + temporal conv + temporal attention

2
Dataset

PEMS08 - California highway traffic sensors

- 170 sensors
- 5-minute intervals, ~17,000 timesteps
- Features: Flow, Occupancy, Speed (mph → km/h)
- Split: (70-15-15)

HZMetro: Hangzhou Metro system

- Pre-split train/val/test data
- Multiple graph types (connectivity, correlation, similarity)
- Features from metro operations
- Split: (18,2,5)

SHMetro: Shanghai Metro system

- Pre-split train/val/test data
- Multiple graph types (connectivity, correlation, similarity)
- Features from metro operations
- Split: (62, 9, 21)

Dataset Graphical Representation

- Graphical Representation for GCN: Data G = (V,E,A)

Where,

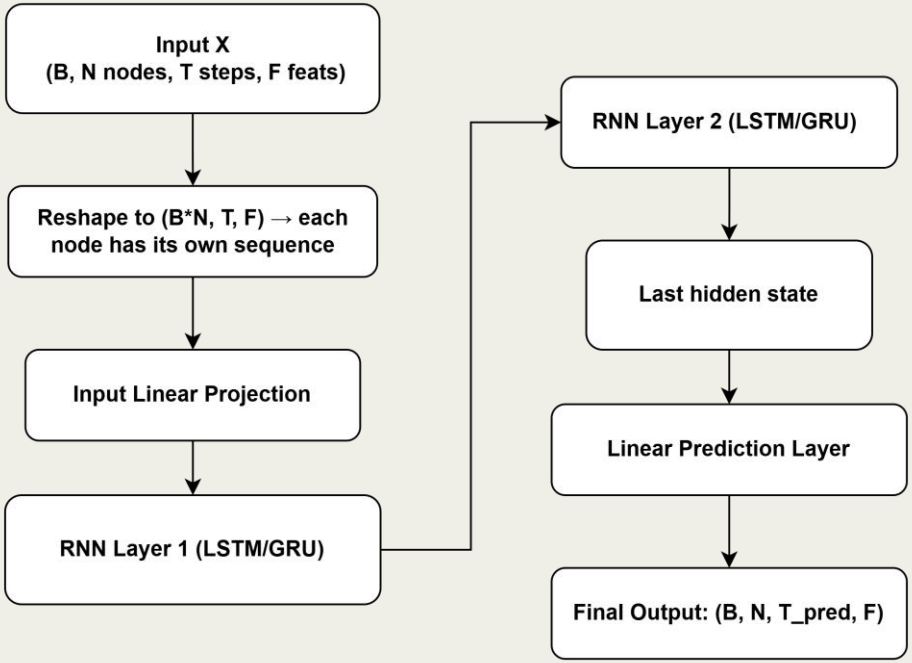
- V = set of sensors # 170
- E = Edges # Between connected sensors
- A = Adjacency Matrix # Weight between corresponding sensors(Distance Decay)

- Each Vertex has 12 * f(number of features per sensor)

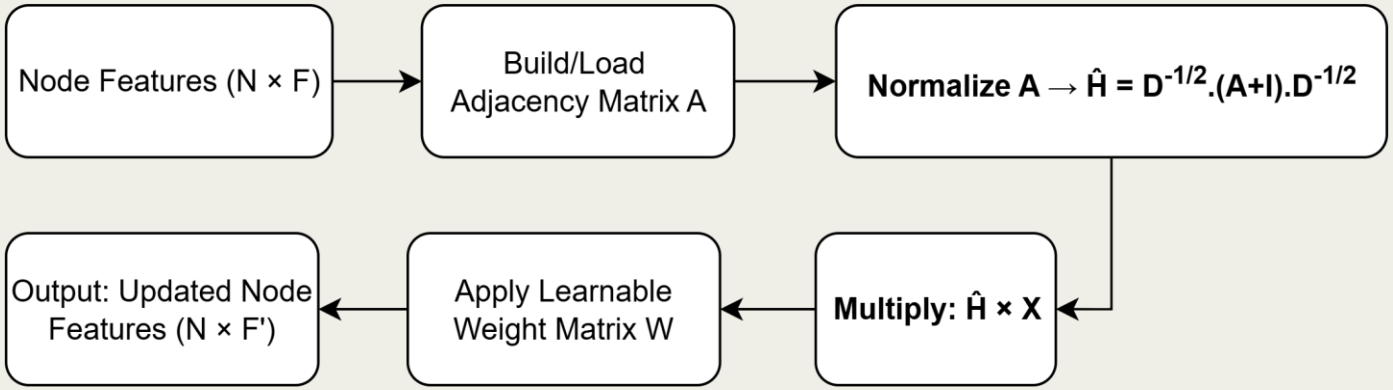
For Example: In Pems08 we have 3 features per sensor so each vertex is 12 * 3 * 1

3
Architecture

RNN/LSTM


Input X
(B, N nodes, T steps, F feats)
Reshape to (B*N, T, F) → each node has its own sequence
Input Linear Projection
RNN Layer 1 (LSTM/GRU)
RNN Layer 2 (LSTM/GRU)
Last hidden state
Linear Prediction Layer
Final Output: (B, N, T_pred, F)

What is Graphical Convolution?



Graph Convolution vs Image Pixel Convolution?

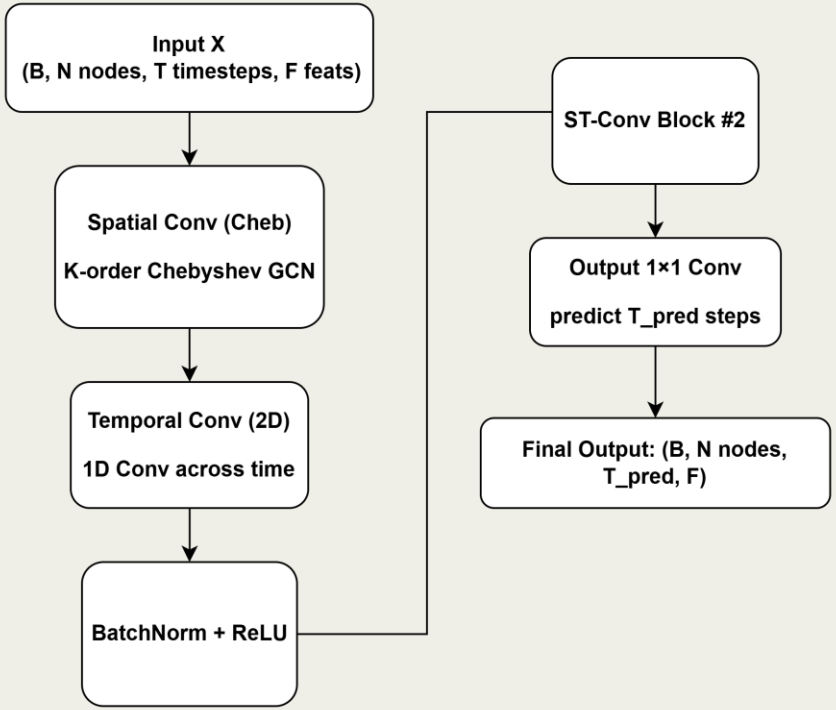
Image Pixel Convolution:

- Operates on grid-structured data (2D fixed neighbors)
- Uses fixed kernels sliding over the image
- Locality defined by pixel positions

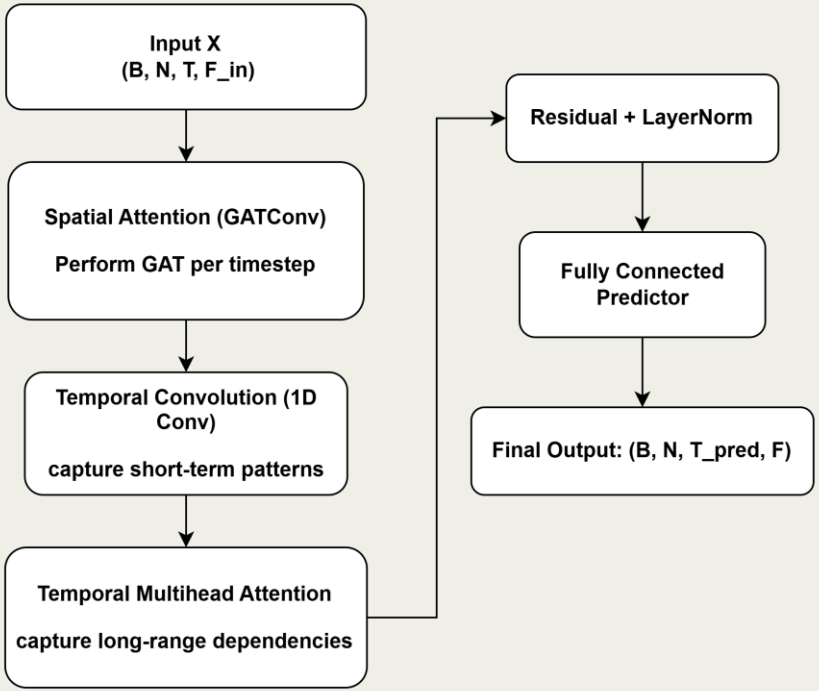
Graph Convolution:

- Operates on irregular graph-structured data
- Neighborhood defined by adjacency matrix

STGCN


Input X
(B, N nodes, T timesteps, F feats)
Spatial Conv (Cheb)
K-order Chebyshev GCN
Temporal Conv (2D)
1D Conv across time
BatchNorm + ReLU
ST-Conv Block #2
Output 1×1 Conv
predict T_pred steps
Final Output: (B, N nodes, T_pred, F)

STGAT


Input X
(B, N, T, F_in)
Spatial Attention (GATConv)
Perform GAT per timestep
Temporal Convolution (1D Conv)
capture short-term patterns
Temporal Multihead Attention
capture long-range dependencies
Residual + LayerNorm
Fully Connected Predictor
Final Output: (B, N, T_pred, F)

4
Metrics

Training Process

For Pems08,
This training was done to predict for 3 timestamps at interval of 5 minutes till 15 minutes for each individual input feature.

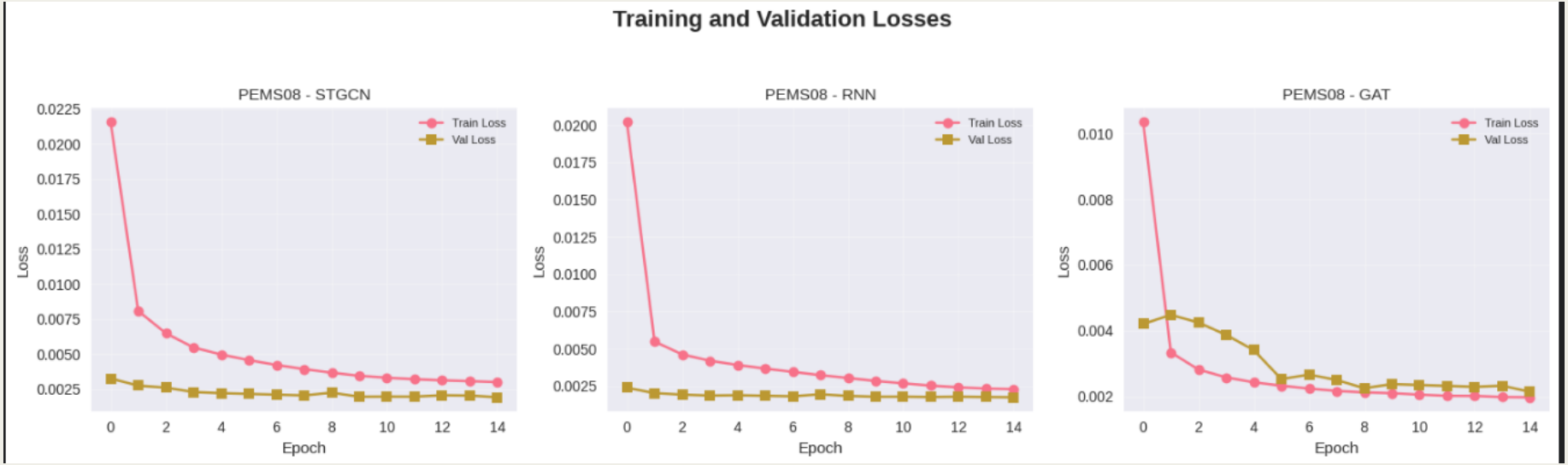
```
============================================================
Training on PEMS08
============================================================

============================================================
Training STGCN Model
============================================================
Epoch [1/15]: Train Loss: 0.021563 | Val Loss: 0.003261 | Best Val Loss: 0.003261 | LR: 0.001000
Epoch [10/15]: Train Loss: 0.003452 | Val Loss: 0.001956 | Best Val Loss: 0.001956 | LR: 0.001000
----------------------------------------
STGCN training completed!
Best validation loss: 0.001918

============================================================
Training RNN Model
============================================================
Epoch [1/15]: Train Loss: 0.020187 | Val Loss: 0.002377 | Best Val Loss: 0.002377 | LR: 0.001000
Epoch [10/15]: Train Loss: 0.002832 | Val Loss: 0.001771 | Best Val Loss: 0.001771 | LR: 0.001000
----------------------------------------
RNN training completed!
Best validation loss: 0.001738

============================================================
Training GAT Model
============================================================
Epoch [1/15]: Train Loss: 0.010339 | Val Loss: 0.004199 | Best Val Loss: 0.004199 | LR: 0.001000
Epoch [10/15]: Train Loss: 0.002093 | Val Loss: 0.002370 | Best Val Loss: 0.002244 | LR: 0.001000
----------------------------------------
GAT training completed!
Best validation loss: 0.002135
```

Training Process



5
Results

PEMS08 Results

•**GAT**: learns edge-wise importance → helpful when adjacency is informative but weights should vary with state. More expressive, so it can fit complex spatial patterns.

•**STGCN**: fixed spectral/localized filters (ChebConv) + temporal convs — good for stationary spatial structure and short-term temporal patterns, but less flexible than attention for dynamic edges.

•**RNN**: excellent at capturing long-term temporal dependencies per node but ignores spatial message passing — works when spatial structure adds little marginal info.

```
================================================================================
For PEMS08
================================================================================

============================================================
COMPREHENSIVE MODEL EVALUATION
============================================================

Evaluating STGCN...
STGCN Results:
  MAE: 6.17 vehicles/5min
  RMSE: 14.79 vehicles/5min
  Best Val Loss: 0.001918

Evaluating RNN...
RNN Results:
  MAE: 5.93 vehicles/5min
  RMSE: 14.35 vehicles/5min
  Best Val Loss: 0.001738

Evaluating GAT...
GAT Results:
  MAE: 6.57 vehicles/5min
  RMSE: 15.15 vehicles/5min
  Best Val Loss: 0.002135

================================================================================
MODEL COMPARISON SUMMARY
================================================================================
Model      MAE            RMSE           Val Loss       Rank
--------------------------------------------------------------------------------
RNN        5.93           14.35          0.001738       1
STGCN      6.17           14.79          0.001918       2
GAT        6.57           15.15          0.002135       3
================================================================================
```

Training Process

Dataset differences drive model suitability

SHMetro: seems to benefit from *spatially adaptive* modeling (GAT best). That implies that on SHMetro the **spatial relationships are dynamic or non-uniform** — some edges/nodes matter more at different times, and attention helps pick those up.

PEMS08: RNN (nodewise sequence model) wins — implying **temporal patterns dominate** and spatial coupling is either weaker, more uniform, or the provided adjacency is less informative. In that case, treating each node independently (but with strong temporal modeling) works better.

```
================================================================================
For SHMetro
================================================================================

============================================================
COMPREHENSIVE MODEL EVALUATION
============================================================

Evaluating STGCN...
STGCN Results:
  MAE: 58.58 vehicles/5min
  RMSE: 133.07 vehicles/5min
  Best Val Loss: 0.006951

Evaluating RNN...
RNN Results:
  MAE: 53.48 vehicles/5min
  RMSE: 121.02 vehicles/5min
  Best Val Loss: 0.006376

Evaluating GAT...
GAT Results:
  MAE: 49.49 vehicles/5min
  RMSE: 104.75 vehicles/5min
  Best Val Loss: 0.005151

================================================================================
MODEL COMPARISON SUMMARY
================================================================================
Model      MAE            RMSE           Val Loss       Rank
--------------------------------------------------------------------------------
GAT        49.49          104.75         0.005151       1
RNN        53.48          121.02         0.006376       2
STGCN      58.58          133.07         0.006951       3
================================================================================
```

6

Limitations & Future Work

Limitations

- GAT is more expressive and can overfit
- Its superior performance on SHMetro suggests it generalizes there but its worse performance on PEMS08 suggests:
  - adjacency info is noisy for PEMS08
  - GAT overfits PEMS08 training but fails to generalize (check train vs val curves).

• STGCN trailing means its temporal receptive field or spatial order K is too small, or the architecture & hyperparameters need tuning.

Conclusion

References

[1] *Spatio temporal Graph Convolutional Network for Multi-Scale Traffic Forecasting – (2021) — Yi Wang and Chang feng Jing*

[2] *LSTM for Traffic Prediction* (2015) — Ma et al.

[3] *STGAT: Spatio-Temporal Graph Attention Network* (2018)

[4] *HZMetro / SHMetro / PEMS08 – DOI*

https://doi.org/10.57760/sciencedb.09286

Thanks!

Do you have any questions?