# Learning to Remember Patterns: Pattern Matching Memory Networks for Traffic Forecasting

Hyunwook Lee, Seungmin Jin, Hyeshin Chu, Hongkyu Lim, and Sungahn Ko

Ulsan National Institute of Science and Technology (UNIST)

ICLR 2022

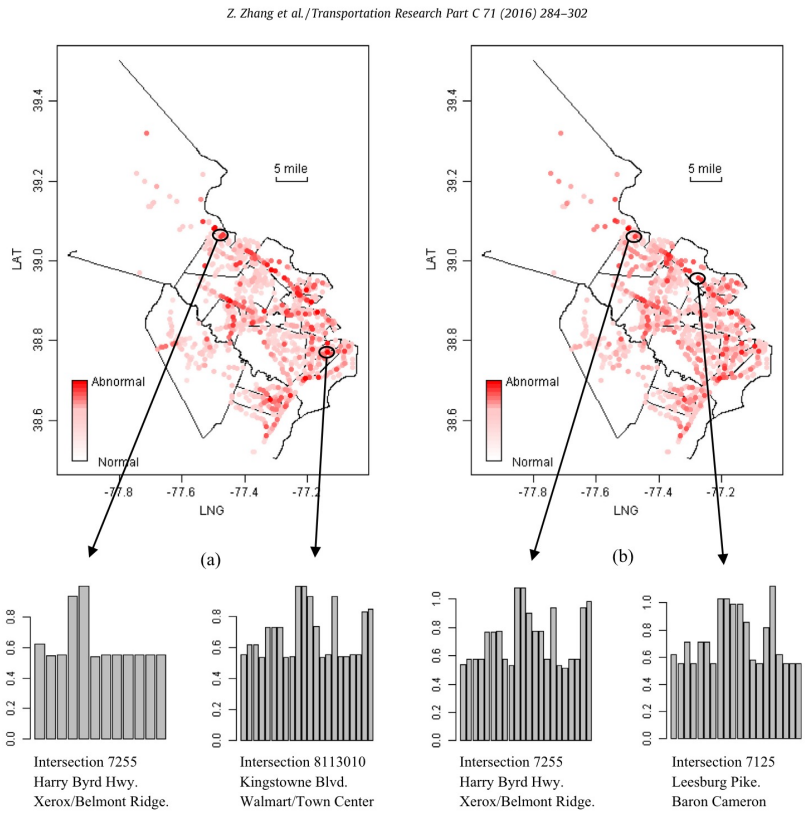
발표자 : 박기연

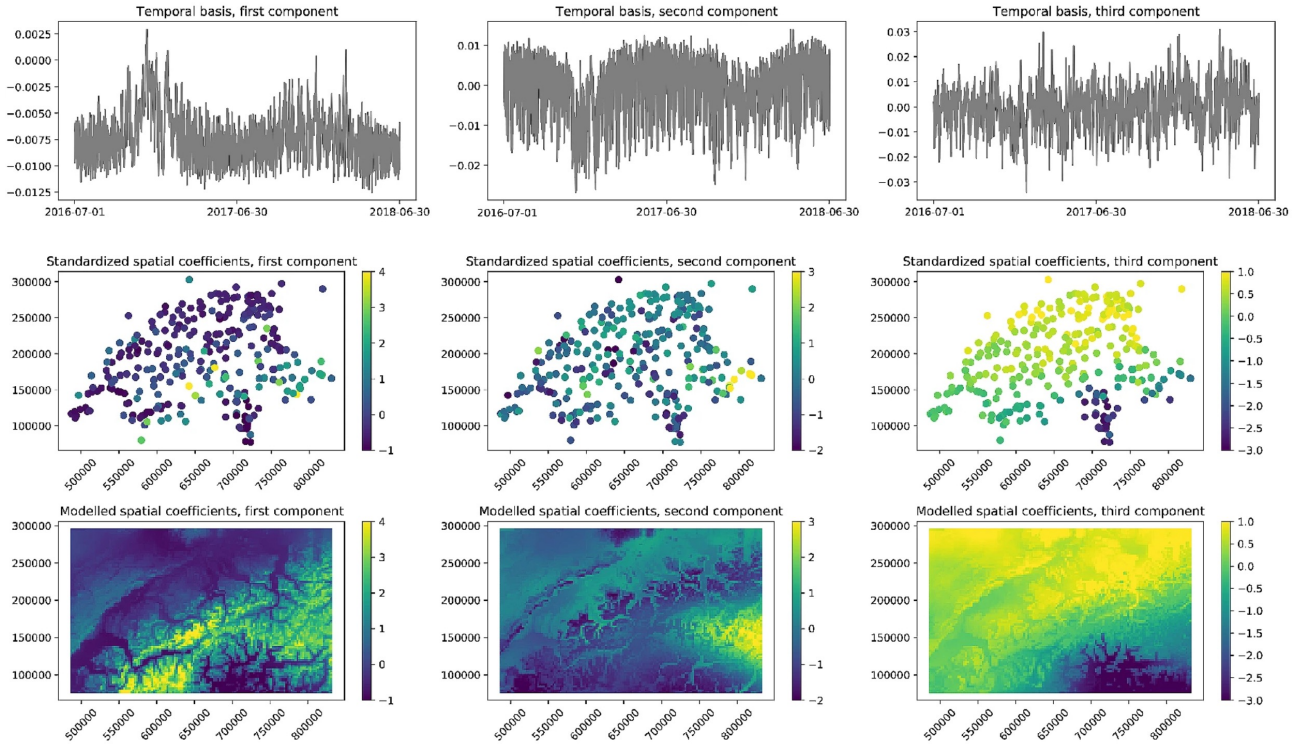
# Index

# Introduction

## Spatial – Temporal Task

- 주어진 시계열 데이터가 여러 개일 경우, 시계열 데이터 간 연관성을 고려한 예측
- 교통 흐름 예측, 기상 예측, 모션 인식의 움직임 예측 등



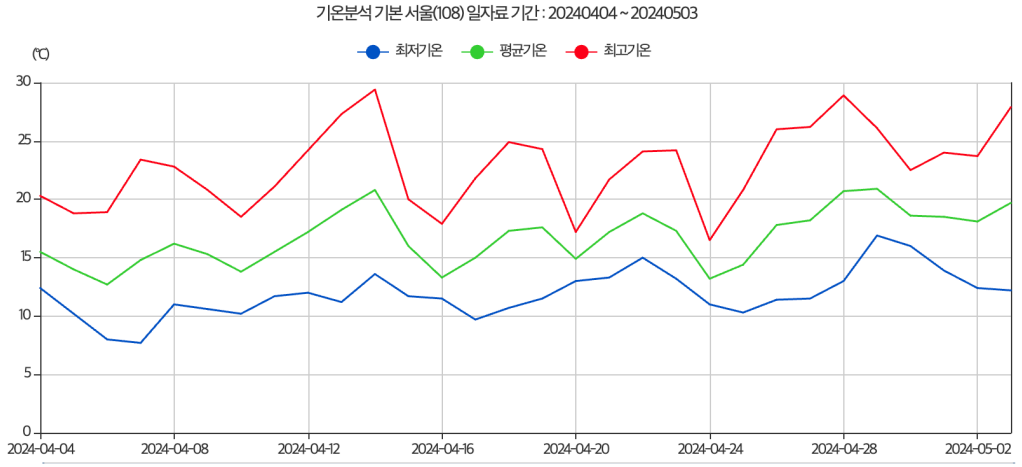



# Introduction

## Temporal Task

- 주어진 시계열 데이터를 토대로 미래 시점의 데이터를 예측
- 주식 예측, 기상 예측, 수요 예측 등


KOSPI 주식 그래프

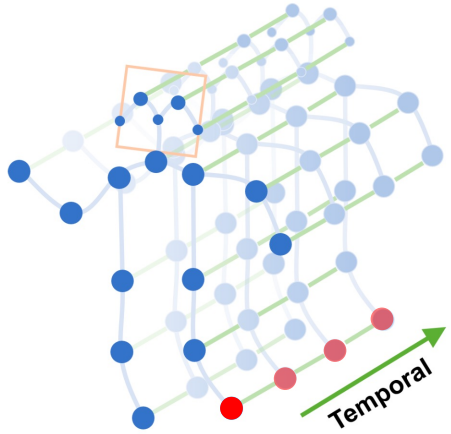



기상청 기온 그래프

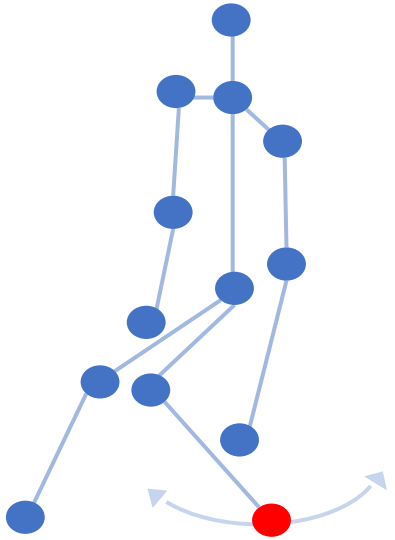
# Introduction

## Temporal Task

- Without Spatial Dependency



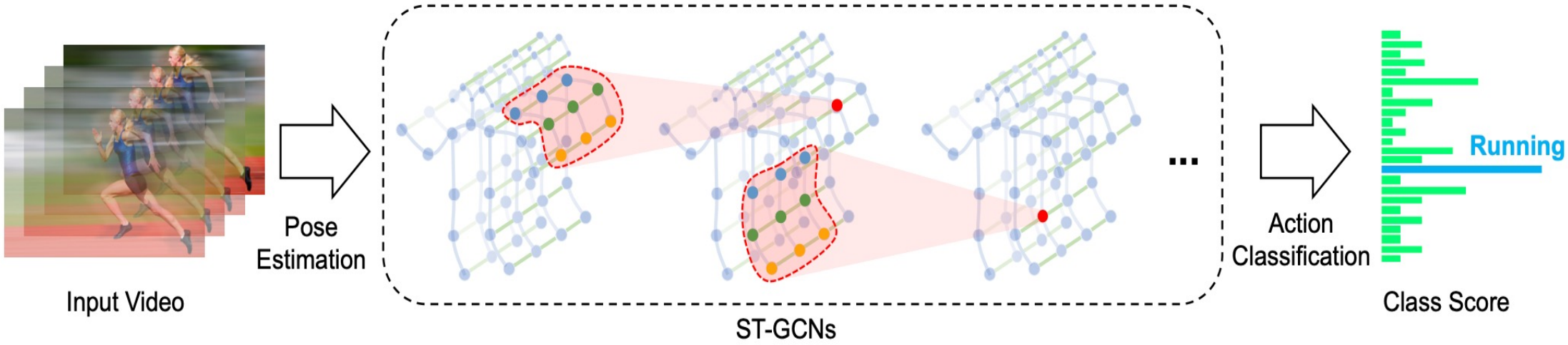

모션 인식



# Introduction

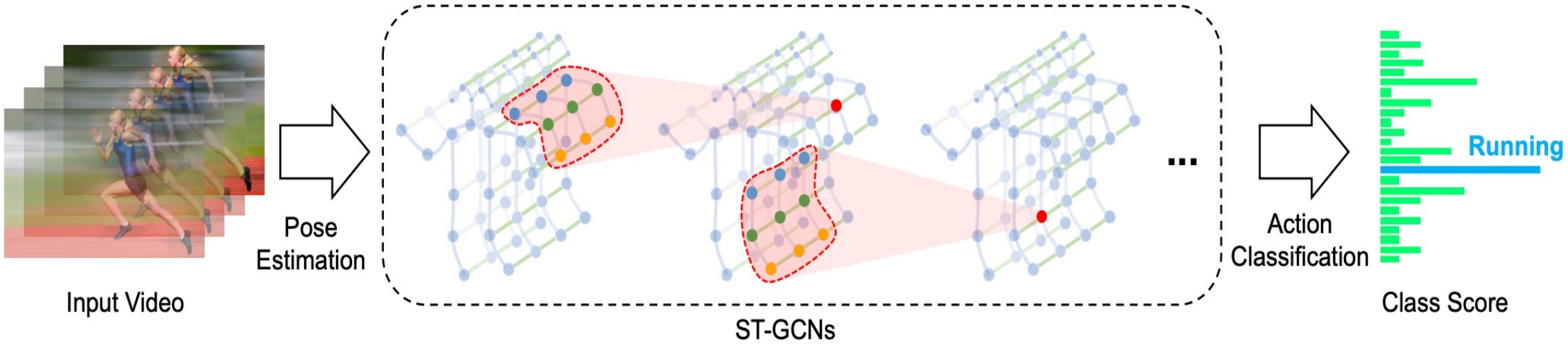
Spatial – Temporal Task

- 주어진 시계열 데이터가 여러 개일 경우, 시계열 데이터 간 연관성을 고려한 예측
- 교통 흐름 예측, 기상 예측, 모션 인식의 움직임 예측 등



# Introduction

## Spatial – Temporal Task




RNN

| time&id | 773869 | 767541 | 767542 | 717447 | 717446 |
|---|---|---|---|---|---|
| 2012-03-01 00:00:00 | 64.375 | 67.625 | 67.125 | 61.5 | 66.875 |
| 2012-03-01 00:05:00 | 62.66666666666670 | 68.55555555555560 | 65.44444444444440 | 62.44444444444440 | 64.44444444444440 |
| 2012-03-01 00:10:00 | 64.0 | 63.75 | 60.0 | 59.0 | 66.5 |
| 2012-03-01 00:25:00 | 57.333333333333300 | 69.0 | 67.66666666666670 | 61.66666666666670 | 67.33333333333330 |
| 2012-03-01 00:30:00 | 66.5 | 63.875 | 67.875 | 62.375 | 64.375 |
| 2012-03-01 00:35:00 | 63.625 | 67.25 | 63.25 | 60.5 | 57.375 |
| 2012-03-01 00:40:00 | 68.75 | 65.25 | 63.5 | 63.0 | 65.125 |
| 2012-03-01 00:45:00 | 63.5 | 61.5 | 62.5 | 58.125 | 66.625 |

Key Point 들의 상관관계 완전히 무시

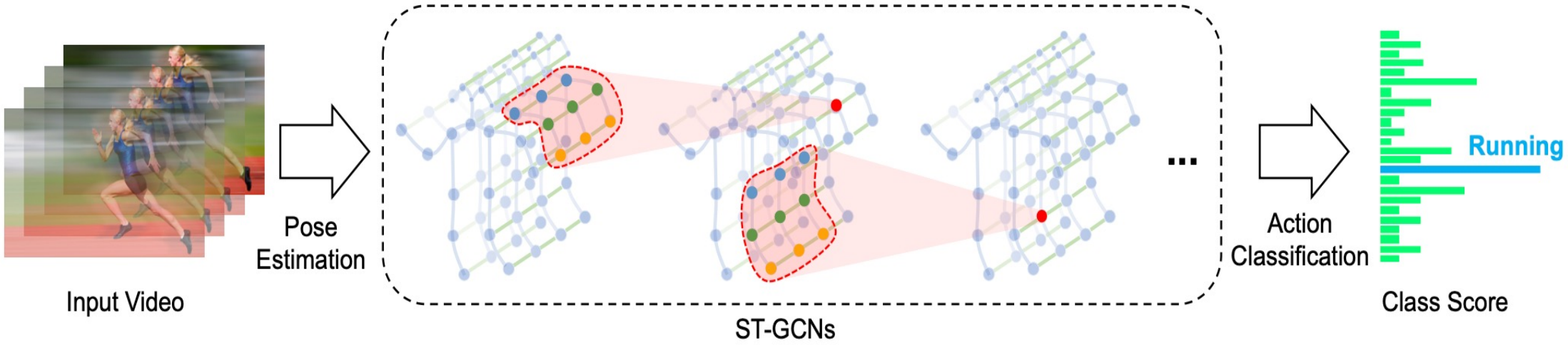
CNN

| time&id | 773869 | 767541 | 767542 | 717447 | 717446 |
|---|---|---|---|---|---|
| 2012-03-01 00:00:00 | 64.375 | 67.625 | 67.125 | 61.5 | 66.875 |
| 2012-03-01 00:05:00 | 62.66666666666670 | 68.55555555555560 | 65.44444444444440 | 62.44444444444440 | 64.44444444444440 |
| 2012-03-01 00:10:00 | 64.0 | 63.75 | 60.0 | 59.0 | 66.5 |
| 2012-03-01 00:25:00 | 57.333333333333300 | 69.0 | 67.66666666666670 | 61.66666666666670 | 67.33333333333330 |
| 2012-03-01 00:30:00 | 66.5 | 63.875 | 67.875 | 62.375 | 64.375 |
| 2012-03-01 00:35:00 | 63.625 | 67.25 | 63.25 | 60.5 | 57.375 |
| 2012-03-01 00:40:00 | 68.75 | 65.25 | 63.5 | 63.0 | 65.125 |
| 2012-03-01 00:45:00 | 63.5 | 61.5 | 62.5 | 58.125 | 66.625 |

실제 Key Point 들의 Spatial 특성 반영 X

# Introduction

## Spatial – Temporal Task




Temporal Model

| time&id | 773869 | 767541 | 767542 | 717447 | 717446 |
|---|---|---|---|---|---|
| 2012-03-01 00:00:00 | 64.375 | 67.625 | 67.125 | 61.5 | 66.875 |
| 2012-03-01 00:05:00 | 62.66666666666670 | 68.55555555555560 | 65.44444444444440 | 62.44444444444440 | 64.44444444444440 |
| 2012-03-01 00:10:00 | 64.0 | 63.75 | 60.0 | 59.0 | 66.5 |
| 2012-03-01 00:25:00 | 57.333333333333300 | 69.0 | 67.66666666666670 | 61.66666666666670 | 67.33333333333330 |
| 2012-03-01 00:30:00 | 66.5 | 63.875 | 67.875 | 62.375 | 64.375 |
| 2012-03-01 00:35:00 | 63.625 | 67.25 | 63.25 | 60.5 | 57.375 |
| 2012-03-01 00:40:00 | 68.75 | 65.25 | 63.5 | 63.0 | 65.125 |
| 2012-03-01 00:45:00 | 63.5 | 61.5 | 62.5 | 58.125 | 66.625 |

Flatten 하게 수집된 데이터 (실제 거리/관련성 무시)

Graph Structure Information

- Feature 상관관계
- Adjacency Matrix
- Feature Attention

# Introduction

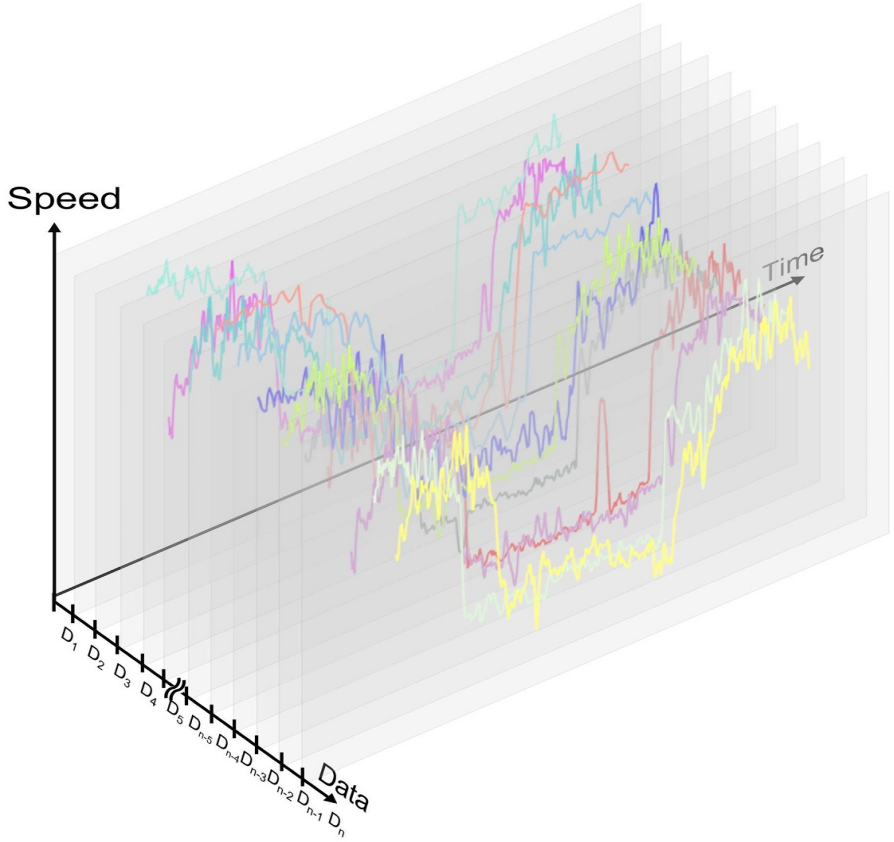
## Traffic Forecasting


Traffic Forecasting Task

- 주요 교통 지점의 Traffic Flow/Speed와 같은 정량적 지표 예측
- Time Sequence 정보 -> 미래의 특정 구간 혹은 Sequence 예측
  (30분 전의 정보 -> 미래 15/30/60분 구간 속도, 교통량 등 예측)
- Traffic Data 수집 지점들은 서로 Spatial Dependency를 가짐
  (특정 도로의 교통량 증가/정체는 다른 도로에 영향을 줄 수 있음)
- 단기 예측 뿐만 아니라, 90분 후 미래와 같은 장기 예측 또한 중요
  (ARIMA, SVM, RNN 과 같은 방법들은 장기 예측에 약점을 보임)

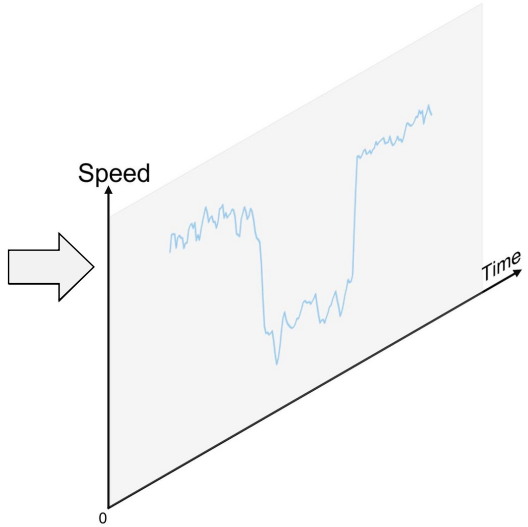
# Introduction

## Traffic Forecasting



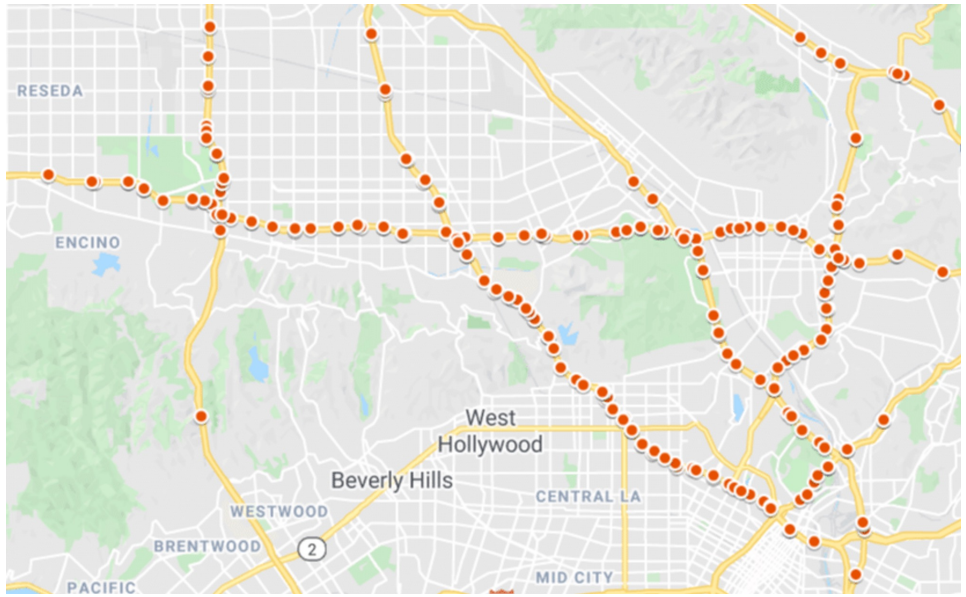

Data Shape

- 각 센서에서의 Time-series 별 속도 측정 값
- Time series 5분 단위로 갱신 (하루 기준 288개의 data 생성)



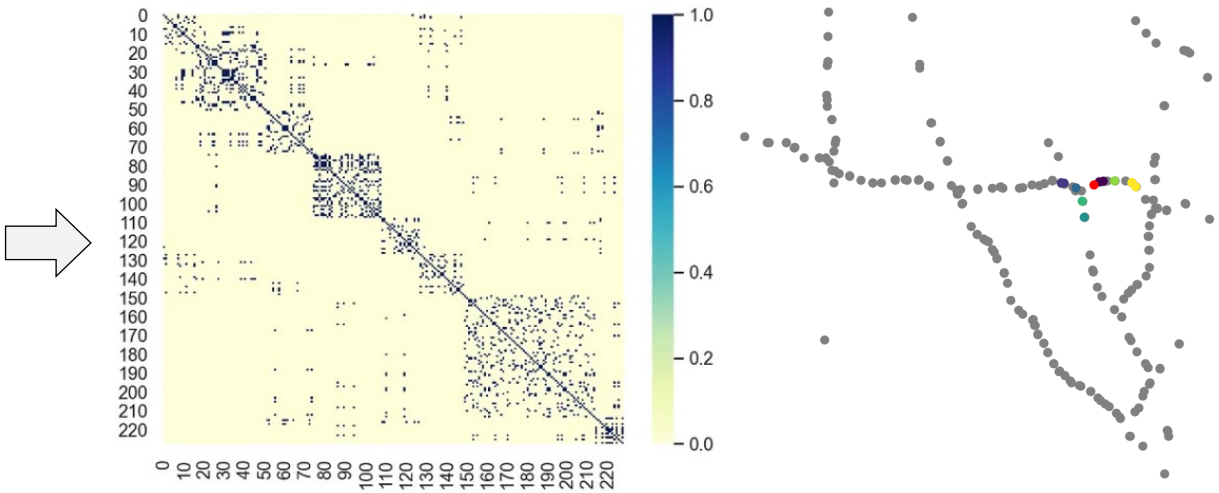
# Introduction

## Traffic Forecasting




Data Shape

- 측정 센서들 간 인접 정보 (Adjacency Matrix)
- 실제 거리를 반영하여 weight 생성
- 일정 거리 이상이면 edge가 생성되지 않는 방식



# Related Work

Traffic Forecasting with ARIMA

- 모수통계학 기반
  - 과거의 시계열 통계 데이터를 활용하여 미래를 예측

- 자기회귀 (AR, Autoregressive)
  - 과거의 값들이 미래 값에 어떤 영향을 미치는 지

- 통합 (I, Integrated)
  - 시계열 데이터를 한 번 이상 차분하여 정상성 확보

- 이동 평균 (MA, Moving Average)
  - 과거 예측 오차가 현재의 예측에 미치는 영향

# Related Work

Traffic Forecasting with ARIMA

- ARIMA(p,d,q)
  - p : 자기회귀(AR) 부분의 차수 [ 과거 몇 개의 시점을 모델링에 포함할 지 ]
  - d : 차분(I) 부분의 차수 [ 데이터를 몇 번 차분할 지 ]
  - q : 이동평균(MA) 부분의 차수 [ 과거 몇 개의 오차 항을 모델링에 포함할 지 ]

- Pros
  - 차분을 통해 비정상 시계열 데이터 예측 가능
  - 유연한 모델링

# Related Work

Traffic Forecasting with ARIMA

- However
  - 교통 데이터는 확률 분포에 대한 엄격한 가정이 불가능함
  - 교통 센서 등의 문제로 결측치, 노이즈 다수 발생
  - 다른 시계열 데이터들에 비해 여러 요인들로 인한 불확실성이 큼
  - ARIMA의 경우 종속 변수만을 가지고 예측하기 때문에 다양한 feature 활용 어려움

- 모수통계학 기반 접근 방식은 실질적으로 교통 데이터에는 적합하지 않다

# Related Work

Traffic Forecasting with Sequence Model (RNN, LSTM, GRU)

- ARIMA의 단점을 보완하기 위해 ARCH, GARCH, VAR, VECM 등의 모델들이 등장
- 하지만 다변량 시퀀스 데이터에 대한 더욱 효과적인 학습 및 예측이 필요
- 기술의 발달로 여러 ANN이 등장

- Why RNN and RNN based Models?
  - 기존 NN 모델의 한계
    - 은닉층에서 활성화 함수를 지난 값은 오직 출력층 방향으로만 향함
    - 기존 NN, CNN 등으로는 시퀀스 데이터에 대한 처리 성능이 낮았음
    - 가중치가 데이터의 처리되는 순서와 상관 없이 업데이트 되기 때문에 이전 샘플 기억 X
    - FFNN(Feed-forward Neural Network)은 입력 길이가 고정되어 있음

# Related Work

Traffic Forecasting with Sequence Model (RNN, LSTM, GRU)

- RNN
  - 시퀀스 길이에 상관 없이 인풋, 아웃풋이 가능해서 다양하게 구조를 만들 수 있음
  - 이전 상태에 대한 정보를 메모리 형태로 저장하여 시계열 데이터를 다룰 때 매우 효과적

- LSTM
  - RNN의 기울기 소실 문제로 인해 교통 흐름에 있어 장기 예측이 불가능 했던 점을 보완
    - Gradient flow를 제어할 수 있는 Gate 구조를 통함
    - 기존의 모델들: 5분~60분 전의 t-1 슬라이드를 통해 5분 후의 t 슬라이드를 예측하는 task가 대다수
  - 하지만 RNN보다 복잡한 계산 구조로 연산 속도가 느림

- GRU
  - LSTM의 복잡한 구조를 단순화, 학습할 파라미터를 줄임

# Related Work

Traffic Forecasting with STGNN

Spatial Models
(CNN-based Approaches)

Temporal Models
(RNN-based Approaches)

| 2016 | 2018 | After |
|---|---|---|
| Structural-RNN | GCRN | Graph-WaveNet |
| | DCRNN | ASTGCN |
| | CGCN | Conv-LSTM NN |
| | STGCN | DGSTNN |
| | GaAN | . |
| | | . |
| | | . |

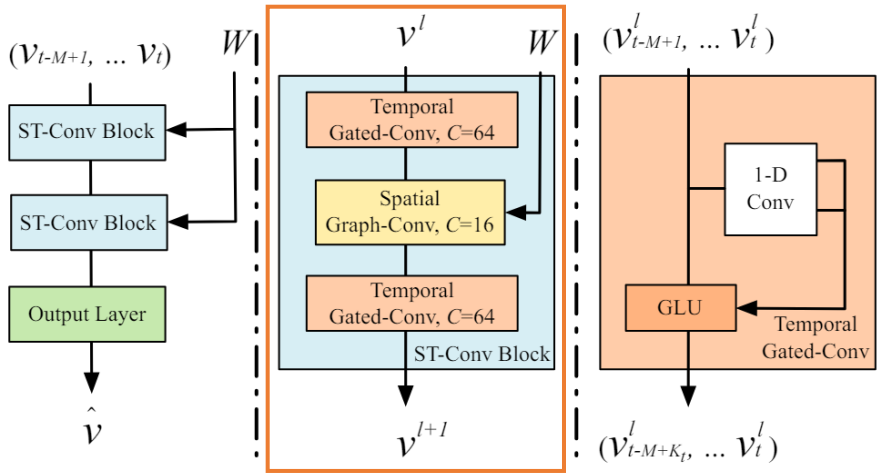
# Method

## Problem Definition

- 시간에 따라 Spatial Importance 달라지지만, 고정된 adjacency matrix가 사용됨
  - 이를 해결하기 위해 GAT, GWNet 같은 모델은 learnable한 인접행렬을 채택하기도 함

- Temporal -> Spatial -> Temporal 과 같은 순차적인 fusion 구조

STGCN Architecture

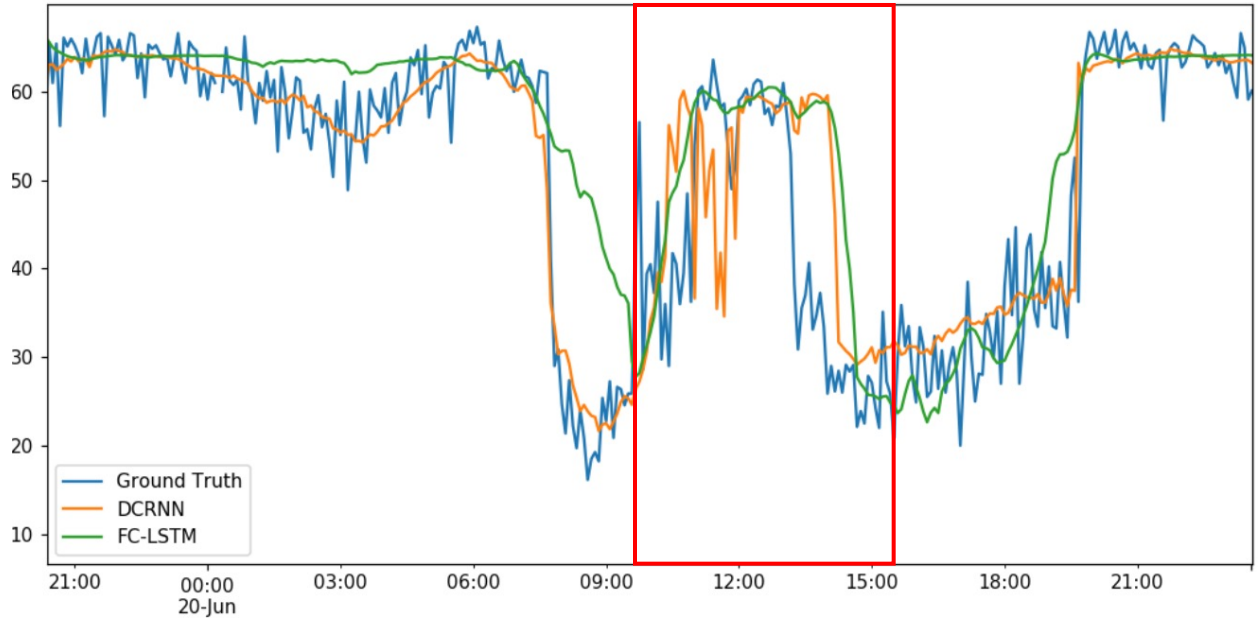



- Fusion이 순차적으로 이루어짐에 따라 정보의 cascading 가능성
- 순차 적용 대신, 동시 적용할 수 있는 모델이 필요

# Method

## Problem Definition

- 갑작스런 속도 변화에 대해 불안정한 예측 성능을 보임
  - Rush hour, Accidents, etc…
  - 장기적인 예측에 있어 문제를 일으키는 원인
  - 갑작스런 속도 변화에 Robust한 방법 필요



# Methods

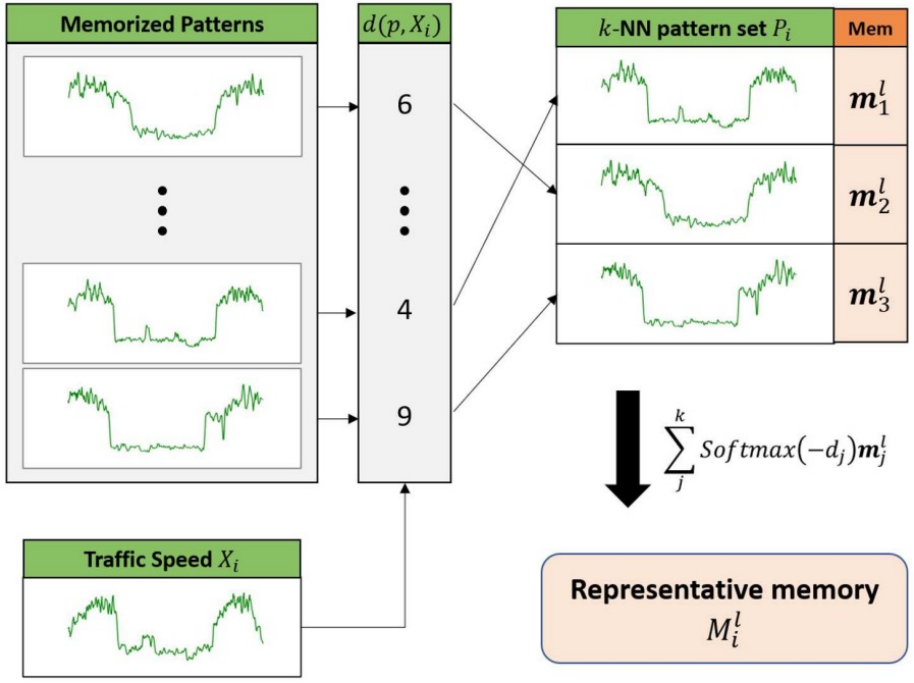
Proposal

## Pattern Matching Memory Networks

GCN + Attention Mechanism + Learnable Adjacency Matrix + Mem2seq

# Method

## Pattern Matching



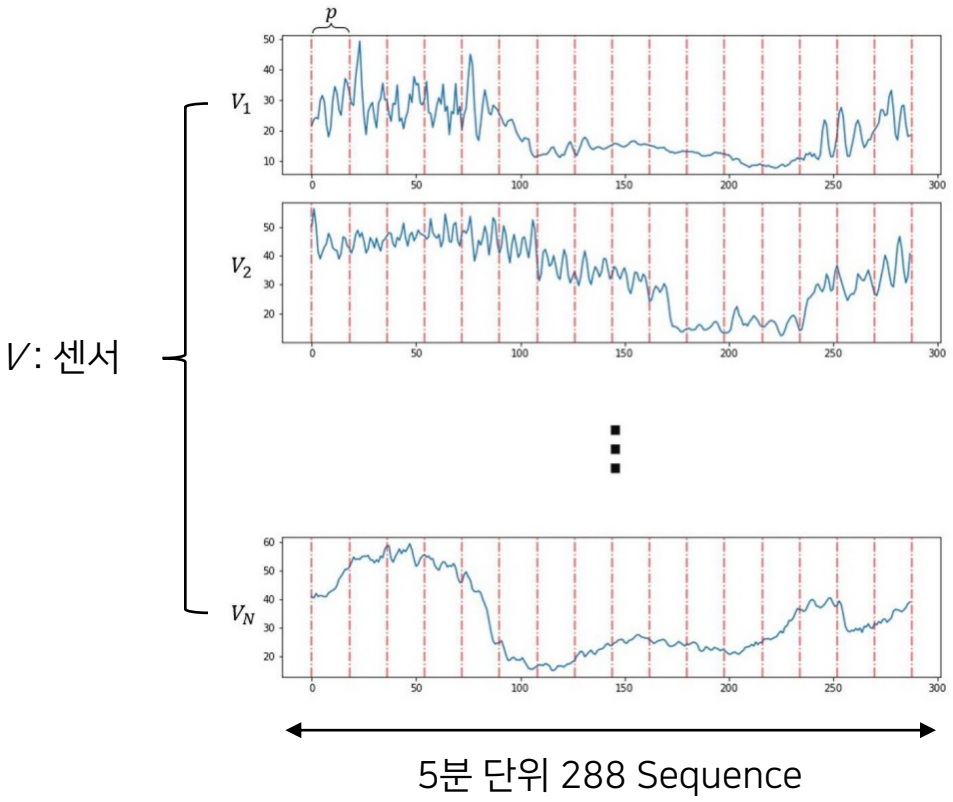

- Input Data를 그대로 모델에 주입 X
- Memory Slot에서 유사 패턴을 추출하여 input을 표현

$$\left[X_{\mathcal{G}}{}^{(t-T'+1)}, \ldots, X_{\mathcal{G}}{}^{(t)}\right] \xrightarrow{f(\cdot)} \left[X_{\mathcal{G}}{}^{(t+1)}, \ldots, X_{\mathcal{G}}{}^{(t+T)}\right]$$

$$\left[X_{\mathcal{G}}{}^{(t-T'+1)}, \ldots, X_{\mathcal{G}}{}^{(t)}\right] \xrightarrow{d(\cdot),k-NN} \left[P_1^t, \ldots, P_N^t\right] \xrightarrow{f(\cdot)} \left[X_{\mathcal{G}}{}^{(t+1)}, \ldots, X_{\mathcal{G}}{}^{(t+T)}\right]$$

# Method

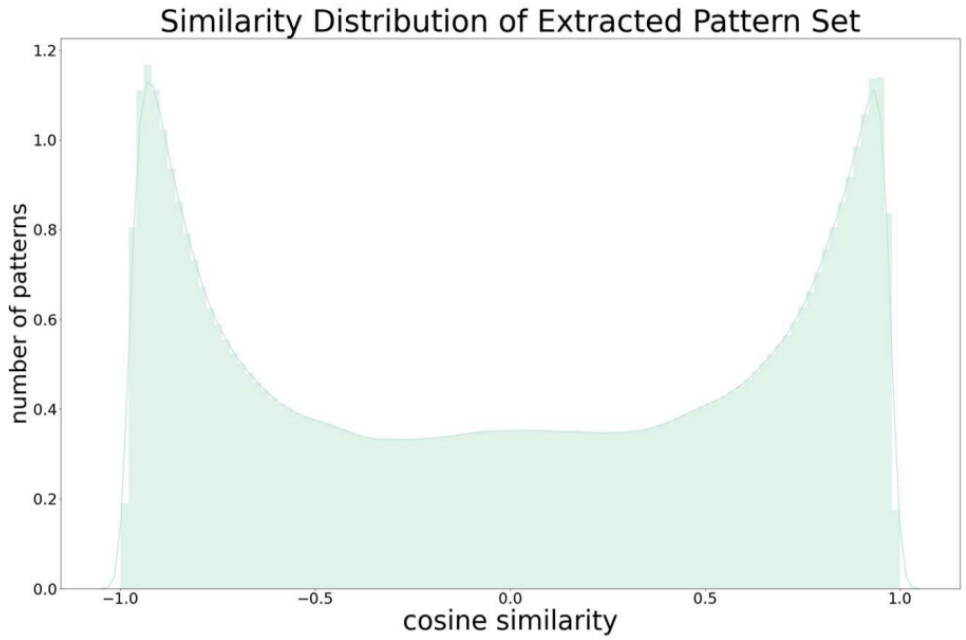
## Key Extraction from Traffic Patterns




- 288개의 time series에 대해 센서 별 하루 평균 패턴을 계산
- 각 센서 마다 대표 패턴 $\mathbb{P}$를 추출함
- Slide Window $T'$ 기준 짧은 패턴 $\mathscr{p}$ 추출
- $|\mathbb{P}| = N \times \left[\frac{288}{T'}\right]$ (즉, 패턴의 개수는 센서 수 N X 센서 별 짧은 패턴 $\mathscr{p}$ 수)

# Method

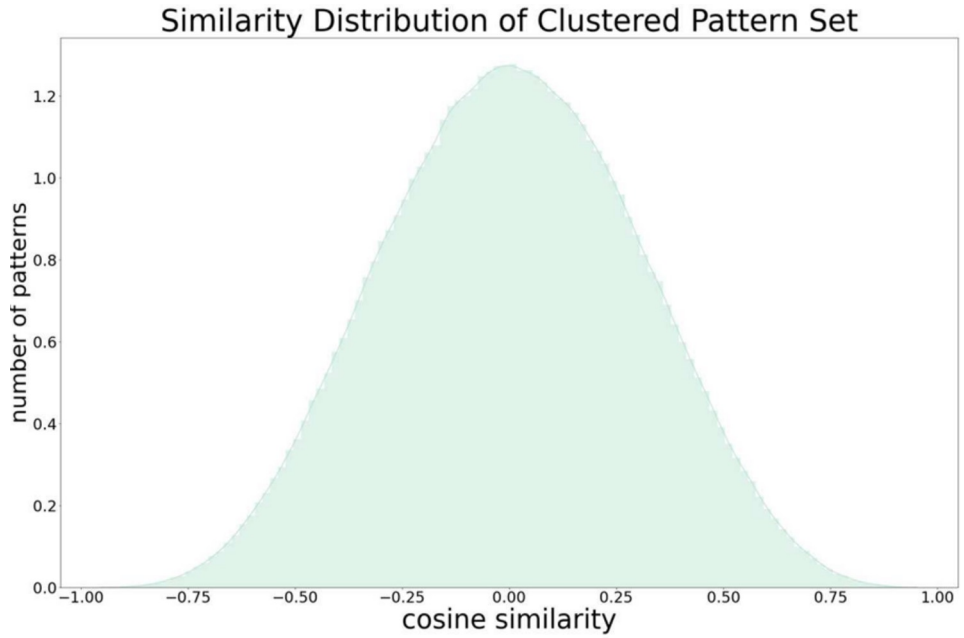
## Pattern Clustering




- 추출된 패턴들의 Cosine 유사도 측정 결과 편향된 분포를 보임
  (유사한 패턴들이 더 많이 존재 = 클래스 불균형)
  - 메모리 비효율성
  - 편향된 학습 결과

# Method

## Pattern Clustering

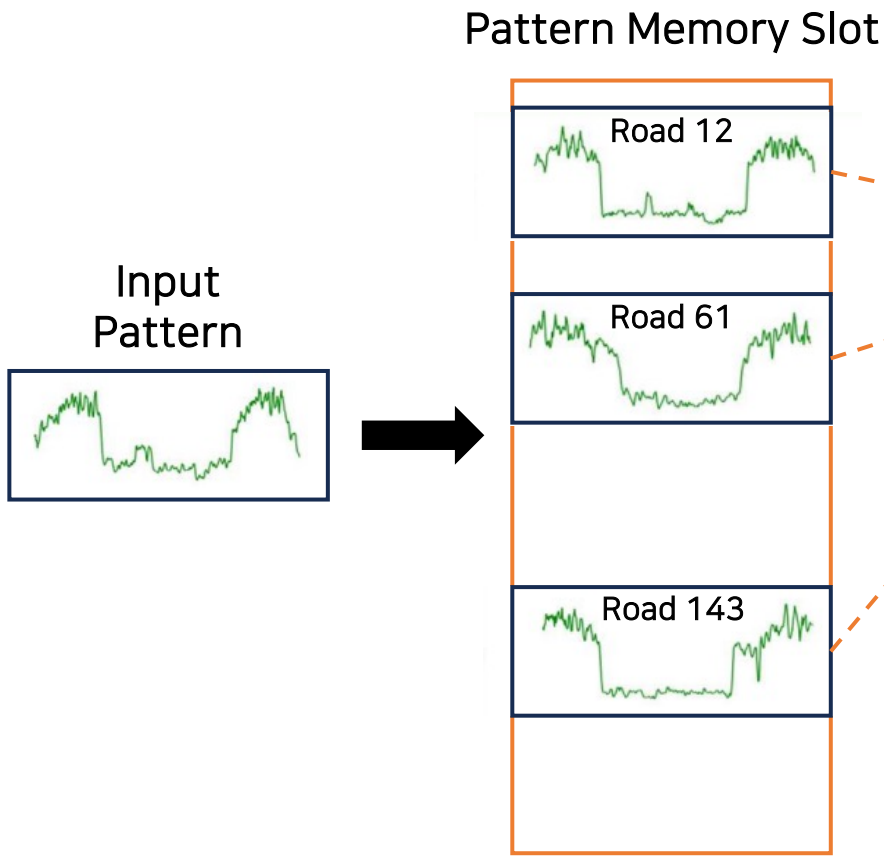



- 패턴들의 Balance 조절을 위해 Clustering-based undersampling
  - 패턴 $\mathcal{p}$와 $\mathcal{p}'$의 Cosine Similarity 측정
  - 유사도가 임계점 $\delta$을 넘기는 경우 같은 클러스터에 배치
  - 클러스터의 center 값만 해당 클러스터의 대표 패턴으로 사용

(실험을 통한 최적의 hyper parameter를 찾아야 함)

# Method

## Neural Memory Architecture




**단순 Sequence 기반으로 Memory Attention 사용**

- 각 패턴은 각기 다른 N개의 Road Sensor에서 추출
- Attention에서 Road Sensor간 상관관계 고려되지 않음

Attention 계산 시 Graph Structure 정보 반영되는 Score 계산 필요

# Method

## Neural Memory Architecture

### Way 1

실제 도로 간 거리 정보

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 1 | 0.0 | 0.0 | 2504.6 | 8563.8 | 8572.5 | 9561.0 | 9590.0 | 5926.3 |
| 2 | 0.0 | 2504.6 | 0.0 | 9293.6 | 9111.3 | 9148.2 | 9177.1 | 4436.7 |
| 3 | 0.0 | 8563.8 | 9293.6 | 0.0 | 1767.4 | 6068.8 | 5401.5 | 9312.3 |
| 4 | 0.0 | 8572.5 | 9111.3 | 1767.4 | 0.0 | 4464.0 | 6985.0 | 9113.1 |
| 5 | 0.0 | 9561.0 | 9148.2 | 6068.8 | 4464.0 | 0.0 | 4654.8 | 9102.4 |
| 6 | 0.0 | 9590.0 | 9177.1 | 5401.5 | 6985.0 | 4654.8 | 0.0 | 0.0 |
| 7 | 0.0 | 5926.3 | 4436.7 | 9312.3 | 9113.1 | 9102.4 | 0.0 | 0.0 |

$$A_{i,j} = \exp(-\frac{dist_{i,j}^2}{2\sigma^2})$$

가우시안 커널

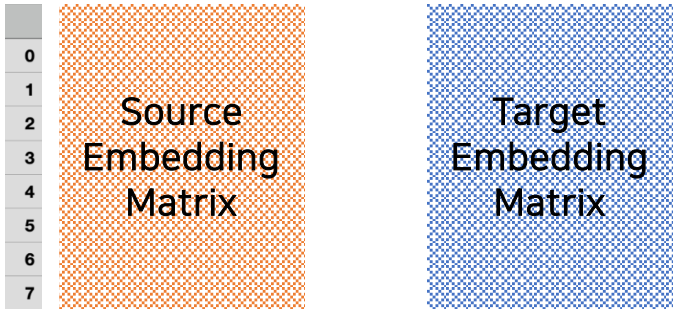
Adjacency Matrix
(edge weighted)

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 1 | 0.0 | 1.0 | 0.3909554 | 0.0 | 0.0 | 0.0 | 0.0 | 0.39045706 |
| 2 | 0.0 | 0.7174379 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 3 | 0.0 | 0.0 | 0.0 | 1.0 | 0.63372165 | 0.0 | 0.0 | 0.0 |
| 4 | 0.0 | 0.0 | 0.0 | 0.62646437 | 1.0 | 0.0 | 0.1351968 | 0.0 |
| 5 | 0.0 | 0.0 | 0.0 | 0.8948124 | 0.36143157 | 1.0 | 0.0 | 0.0 |
| 6 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 |
| 7 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 |

- 거리를 기반으로 weight 부여
- 생성된 weight → attention score에 영향

### Way 2

Dot product

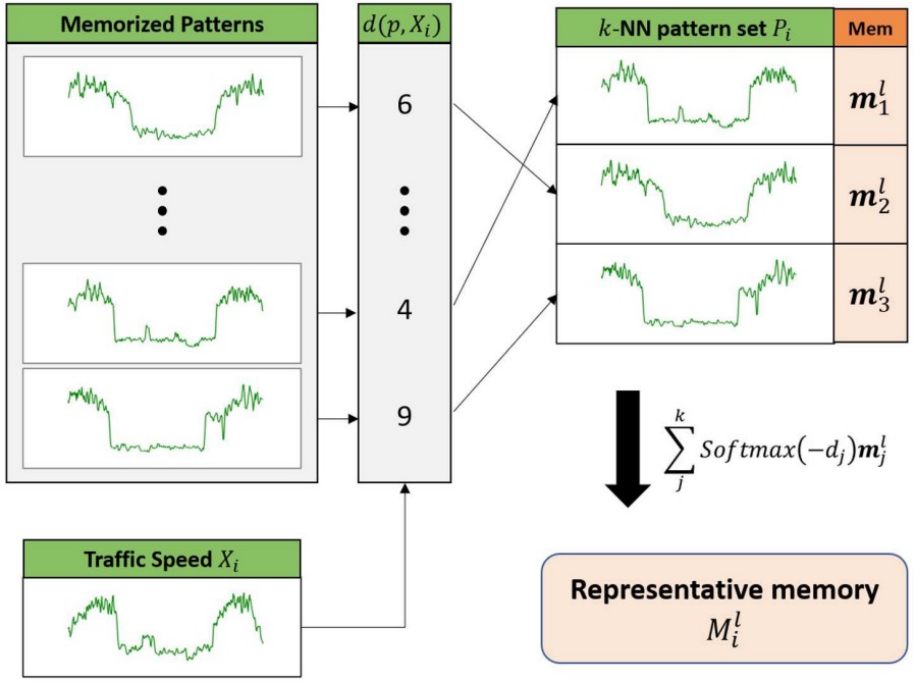



SoftMax(ReLU())

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | Learnable Adjacency Matrix | | | | | | | |
| 1 | | | | | | | | |
| 2 | | | | | | | | |
| 3 | | | | | | | | |
| 4 | | | | | | | | |
| 5 | | | | | | | | |
| 6 | | | | | | | | |
| 7 | | | | | | | | |

# Method

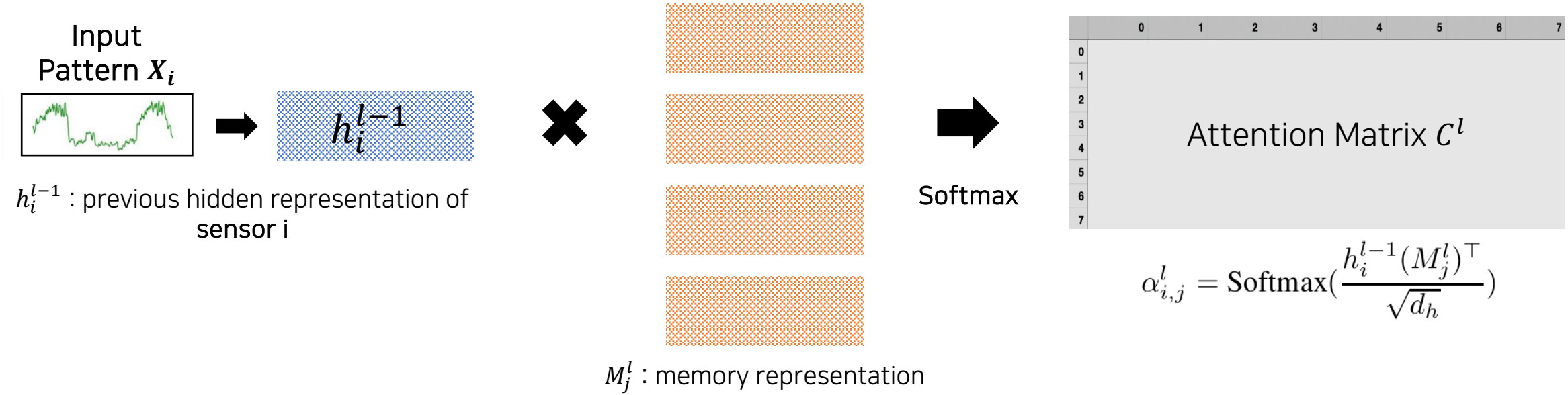
## Neural Memory Architecture




- Input X와 Memory Set의 패턴들 간 거리를 계산
- K-NN (k=3) 을 기준으로 유사 패턴 k개를 추출
- 추출된 k개의 패턴은 embedding matrix를 통해 hidden representation(m) 으로 변환
- 거리가 가까운 것에 가중치 부여
- 패턴들의 가중 합을 통해 메모리를 통한 대표 Representation 생성 (Representative Memory)

# Method

## Neural Memory Architecture



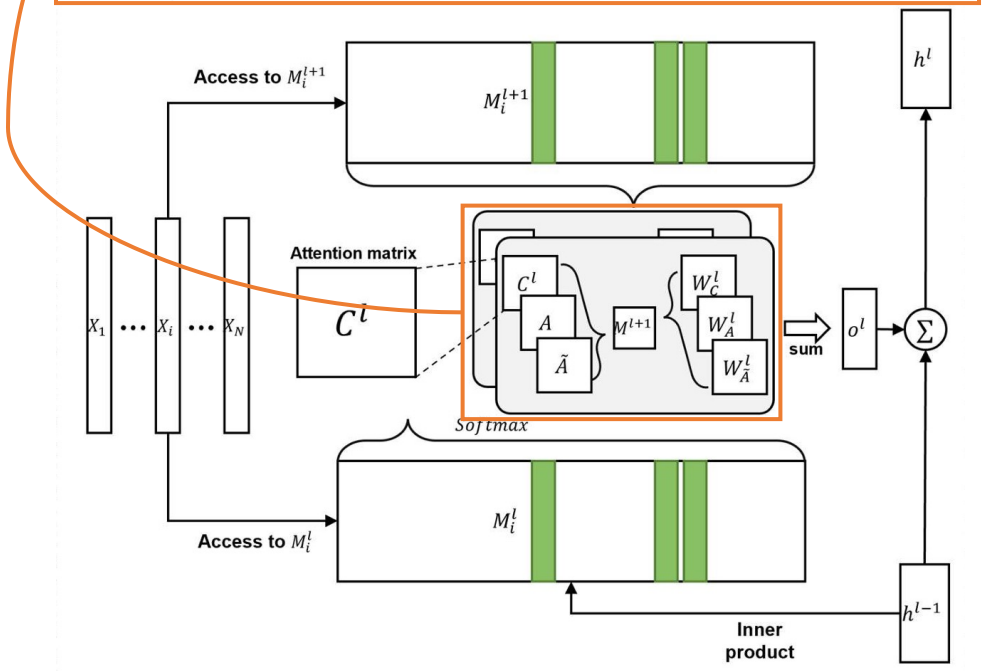

$$\alpha_{i,j}^{l} = \text{Softmax}(\frac{h_i^{l-1}(M_j^l)^\top}{\sqrt{d_h}})$$

- Representation(h)와 Memory Representation을 내적
- Softmax를 취한 값을  Attention Score로 사용
- Memory Representation은 $X_i$ 와 유사한 sensor를 기반으로 생성된 벡터
  - Attention Score는 자신과 거리 상 가장 유사한 sensor를 고려한 Score

# Method

## Neural Memory Architecture

$$o^l = \sum_i^h \left(W^l_{A,i} M^{l+1} A + W^l_{\tilde{A},i} M^{l+1} \tilde{A} + W^l_{C,i} M^{l+1} C^l\right)$$

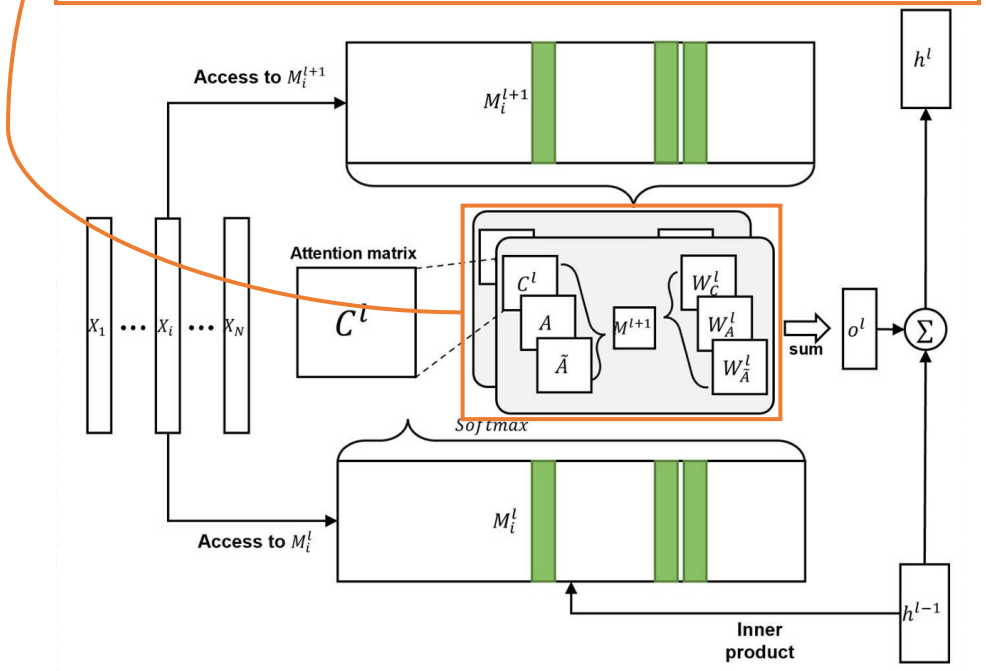



- GCN의 그래프 컨볼루션 식과 유사 $\boldsymbol{f(H^{(l)}, A) = \sigma(AH^{(l)}W^{(l)})}$
  - Original Adjacency Matrix
  - Learnable Adjacency Matrix
  - Attention Matrix
    개별 적용
  - $\tilde{A} = SoftMax(ReLU(E_1 E_2^T))$
    - $E_1$ : Source Node Embedding
    - $E_2$ : Target Node Embedding
    - Learnable Embedding Matrix들을 통한 학습

# Method

## Neural Memory Architecture

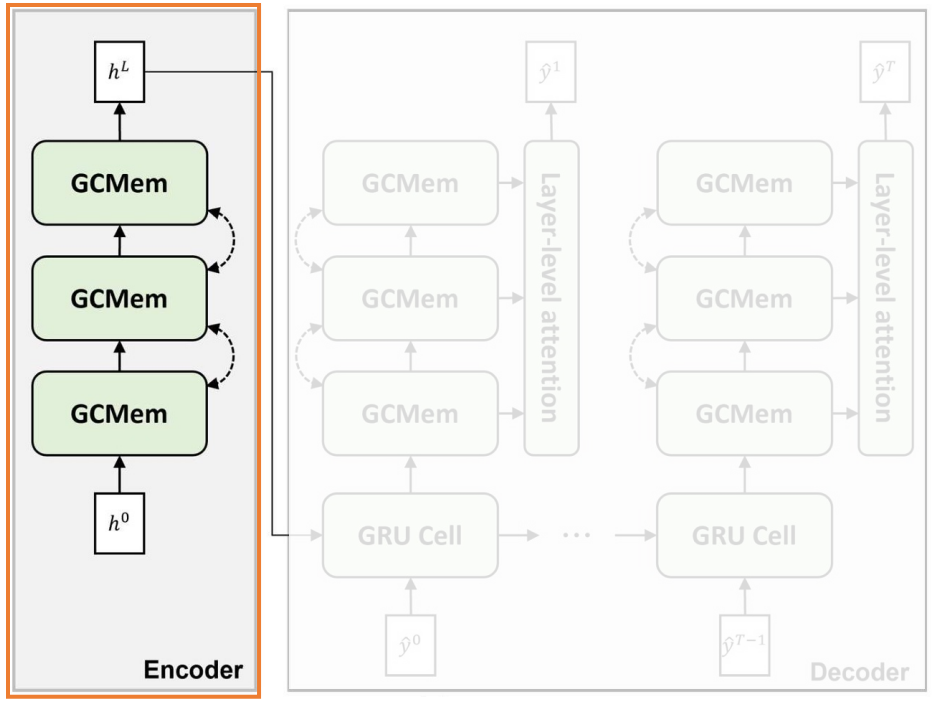
$$o^l = \sum_i^h \left(W^l_{A,i} M^{l+1} A + W^l_{\tilde{A},i} M^{l+1} \tilde{A} + W^l_{C,i} M^{l+1} C^l\right)$$




- GCN의 그래프 컨볼루션 식과 유사 $\boldsymbol{f(H^{(l)}, A) = \sigma(AH^{(l)}W^{(l)})}$
  - 생성되는 $o^l$ 은 3개의 weight를 통해 학습
    - Graph Structure : A
    - Learnable Graph Structure : $\tilde{A}$
    - Pattern Attention : C
  - Update : Memory Cell Vector ($o^l$) + hidden vector ($h^{l-1}$)

# Method

## Encoder Architecture




- 패턴 메모리에서 제공되지 못하는 정보
  - 산업 단지 주변 도로의 고유한 주기성
  - 다양한 이벤트로 인한 잡음 및 이상 패턴
  - 패턴이 그룹화될 때 미리 캡쳐되거나 인코딩되지 않음

- GCMem 에 대한 input query $h_i^0$ 을 생성
  - Time에 대한 Embedding : $emb(T)$
  - Noise : $N_i$
  - A learnable matrix : $W_n$
    (Learnable embedding for the time of a day)
  - $h_i^0 = emb(T) + W_n N_i$

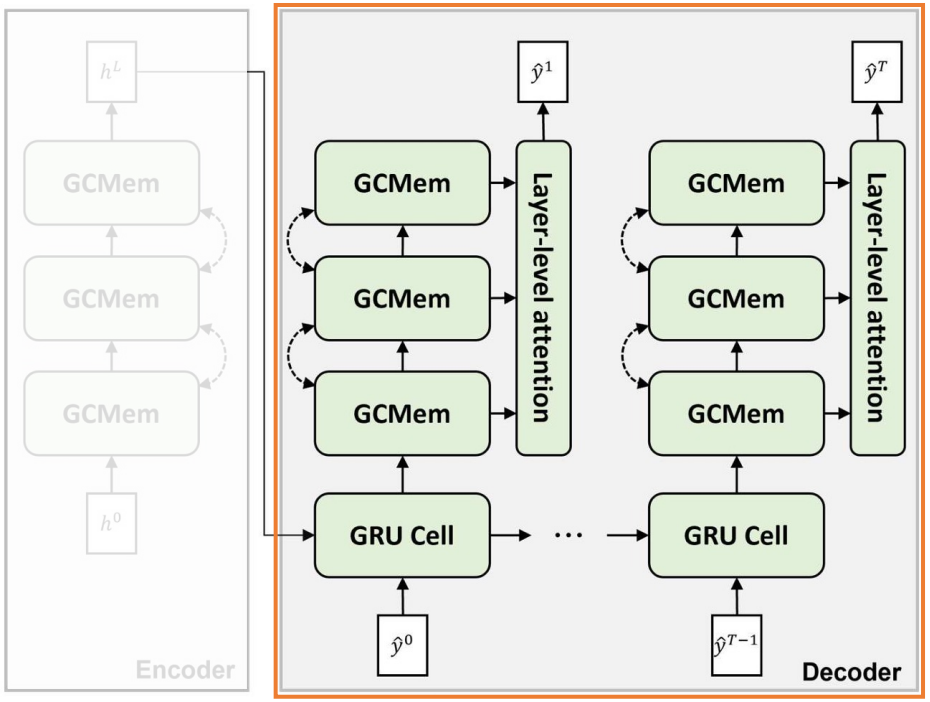
$$\mathrm{T} = [t - T' + 1, \dots, t]$$

$$N_i = X_i - p1$$

- $L$-layer GCMem → $h^L$

# Method

## Decoder Architecture

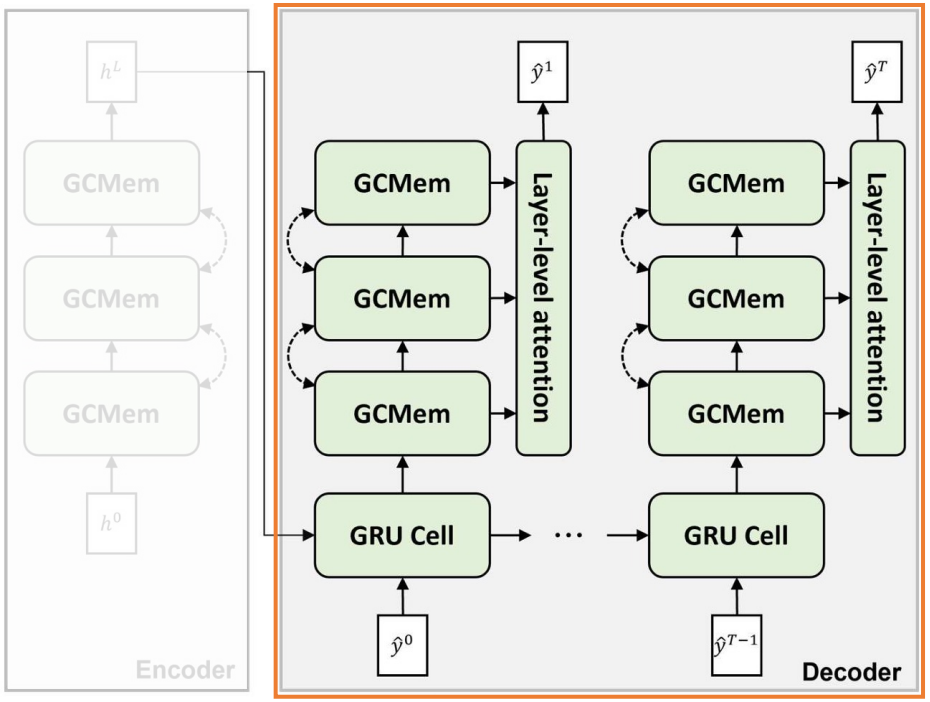



- For each $t$ step, Decoder predicts the value at time step $t$
  - GRU Hidden State $\widetilde{\boldsymbol{h}}_{\boldsymbol{t-1}}$ & 이전 예측 값 $\hat{\boldsymbol{y}}^{\boldsymbol{t-1}}$
    - 초기 $\hat{y}_0$ : Zero Matrix
    - 초기 $\tilde{h}_0$ : Last Encoder Hidden State $h^L$

- 각 GCMem Layer 마다 hidden state와 input을 통해 생성되는 memory representation 유사도 기반으로 attention score 산출

# Method

## Decoder Architecture




- Details

$$\alpha_{i,j} = Softmax(e_{i,j})$$

$$e_{i,j} = \frac{\left(h_i^{l-1}\right)\left(M_i^l W^l\right)^{\mathrm{T}}}{\sqrt{d_h}} \qquad \hat{y}_{t,i} = \sum_l \alpha_{i,j} o_i^l W_{proj}$$

- $h_i^{l-1}$ : Previous Hidden State
- $M_i$ : Memory Representation
- $Softmax(e_{i,j})$를 통해 attention weight (Energy Score) 생성
- GCMem Layer 마다 output에 해당 weight를 통해 가중합 결과 산출

# Experiments

## Datasets & Setup

| DATASET | Naver_서울 | METR-LA |
|---|---|---|
| 수집기간 | **약 3개월**<br>2020-09-02 00:00 ~<br>2020-12-01 23:55 | **약 4개월**<br>2012-03-01 00:00 ~<br>2012-06-27 23:55 |
| 센서 수 | 774 | 207 |
| 특이 사항 | 5분 단위 수집<br><br>복잡한 서울 도심 정보<br>(급격한 변화가 많은 데이터 셋) | 5분 단위 수집 |

$$|\mathbb{P}| = N \times \left\lceil \frac{288}{T'} \right\rceil$$

- Input Sequence 및 Pattern Sequence 길이 : 90분 ($T' = 18$)
- Output Sequence : 90분 (실제 평가는 15/30/60/90 분 시점에서 진행)
- 실제 Pattern 추출 개수 : 12384 (이중 Under Sampling을 통해 약 100 개 산출)

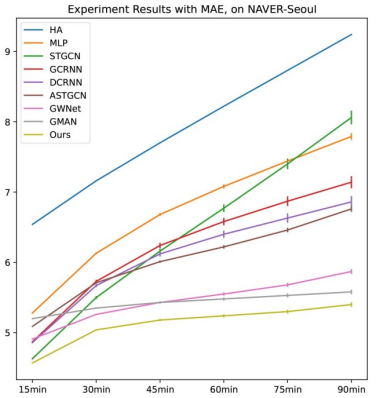
Greedy Search → $d_h = [16, 32, 64, 128]$, $L = [1, 2, 3, 4]$, $|\mathbb{P}| = various\ values\ (\approx 100)$, $\delta = 0.7\ /\ 0.9$

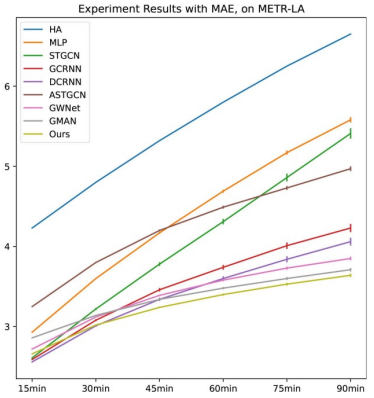
# Experiments

## Evaluation

Table 1: Experimental Results for NAVER-Seoul and METR-LA datasets

| Dataset | T | Metric | HA | MLP | STGCN | GCRNN | DCRNN | ASTGCN | GWNet | GMAN | PM-MemNet |
|---|---|---|---|---|---|---|---|---|---|---|---|
| NAVER-Seoul | 15min | MAE | 6.54 | 5.28 | 4.63 | 4.87 | 4.86 | 5.09 | 4.91 | 5.20 | **4.57** |
| | | MAPE | 18.24 | 16.86 | 14.49 | 15.23 | 15.35 | 16.14 | 14.86 | 16.98 | **14.43** |
| | | RMSE | 9.32 | 7.78 | 6.92 | 7.18 | 7.12 | 7.44 | 7.24 | 8.32 | **6.72** |
| | 30min | MAE | 7.16 | 6.13 | 5.50 | 5.73 | 5.67 | 5.71 | 5.26 | 5.35 | **5.04** |
| | | MAPE | 20.15 | 20.05 | 17.37 | 18.17 | 18.38 | 18.78 | **16.16** | 17.47 | 16.34 |
| | | RMSE | 10.18 | 9.51 | 8.83 | 9.03 | 8.80 | 8.73 | 8.13 | 8.67 | **7.86** |
| | 60min | MAE | 8.22 | 7.08 | 6.77 | 6.58 | 6.40 | 6.22 | 5.55 | 5.48 | **5.24** |
| | | MAPE | 23.37 | 23.44 | 20.42 | 20.95 | 21.09 | 20.37 | 16.97 | 17.89 | **16.94** |
| | | RMSE | 11.54 | 11.13 | 10.89 | 10.58 | 10.06 | 9.58 | 8.77 | 8.94 | **8.39** |
| | 90min | MAE | 9.24 | 7.79 | 8.06 | 7.14 | 6.86 | 6.76 | 5.87 | 5.58 | **5.40** |
| | | MAPE | 26.40 | 26.08 | 22.93 | 22.86 | 22.74 | 21.83 | 17.89 | 18.18 | **17.44** |
| | | RMSE | 12.77 | 12.17 | 12.86 | 11.43 | 10.69 | 10.32 | 9.33 | 9.09 | **8.68** |
| METR-LA | 15min | MAE | 4.23 | 2.93 | 2.61 | 2.59 | **2.56** | 3.25 | 2.72 | 2.86 | 2.66 |
| | | MAPE | 9.76 | 7.76 | **6.59** | 6.73 | 6.67 | 9.27 | 7.14 | 7.67 | 7.06 |
| | | RMSE | 7.46 | 5.81 | 5.19 | **5.12** | 5.10 | 6.28 | 5.20 | 5.77 | 5.28 |
| | 30min | MAE | 4.80 | 3.60 | 3.22 | 3.08 | **3.01** | 3.80 | 3.12 | 3.14 | 3.02 |
| | | MAPE | 11.30 | 10.00 | **8.39** | 8.72 | 8.42 | 11.28 | 8.66 | 8.79 | 8.49 |
| | | RMSE | 8.34 | 7.29 | 6.63 | 6.32 | 6.29 | 7.59 | 6.34 | 6.54 | **6.28** |
| | 60min | MAE | 5.80 | 4.69 | 4.31 | 3.74 | 3.60 | 4.49 | 3.58 | 3.48 | **3.40** |
| | | MAPE | 14.04 | 13.68 | 11.13 | 11.50 | 10.73 | 13.69 | 10.30 | 10.10 | **9.88** |
| | | RMSE | 9.86 | 9.24 | 8.71 | 7.71 | 7.65 | 8.94 | 7.53 | 7.30 | **7.24** |
| | 90min | MAE | 6.65 | 5.58 | 5.41 | 4.23 | 4.06 | 4.97 | 3.85 | 3.71 | **3.64** |
| | | MAPE | 16.37 | 17.08 | 13.76 | 13.49 | 12.53 | 15.53 | 11.39 | 11.00 | **10.74** |
| | | RMSE | 10.97 | 10.52 | 10.47 | 8.79 | 8.58 | 9.71 | 8.12 | 7.71 | **7.74** |




- STGCN, Graph WaveNet : 1D – CONV 기반 모델, 4가지 시점 한번에 산출
- GCRNN, DCRNN : RNN 기반 모델 → Short term 예측에 좋은 성능 도출
- PM-MemNet 역시 GRU 기반 모델이지만 Pattern Memory를 통해 Long term에서 성능 향상
- ASTGCN, GMAN : GAT 기반 Attention 모델

# Experiments

## Evaluation

Table 2: Ablation study result. Note that 'Ours' means PM-MemNet.

| Dataset | T | Metric | Ours | SimpleMem | CNN Decoder | RNN Decoder | Ours (L=1) | Ours ($\|\mathbb{P}\| = k$) | Ours ($\|\mathbb{P}\| >> 1000$) |
|---|---|---|---|---|---|---|---|---|---|
| NAVER-Seoul | 15min | MAE | 4.57 | 5.72 | **4.56** | 4.67 | 4.72 | 4.66 | 4.59 |
| | | MAPE | 14.43 | 18.18 | **14.40** | 14.84 | 14.98 | 14.77 | 14.48 |
| | | RMSE | 6.72 | 8.79 | **6.71** | 6.83 | 6.87 | 6.89 | 6.73 |
| | 30min | MAE | **5.04** | 5.84 | 5.06 | 5.19 | 5.22 | 5.21 | 5.09 |
| | | MAPE | **16.34** | 18.86 | 16.36 | 16.87 | 16.97 | 16.91 | 16.41 |
| | | RMSE | **7.86** | 9.24 | 7.90 | 8.02 | 8.03 | 8.18 | 7.97 |
| | 60min | MAE | **5.24** | 6.38 | 5.32 | 5.47 | 5.52 | 5.53 | 5.36 |
| | | MAPE | **16.94** | 21.42 | 17.19 | 17.91 | 17.97 | 18.05 | 17.19 |
| | | RMSE | **8.39** | 10.08 | 8.51 | 8.69 | 8.70 | 8.92 | 8.67 |
| | 90min | MAE | **5.40** | 6.95 | 5.55 | 5.70 | 5.72 | 5.74 | 5.52 |
| | | MAPE | **17.44** | 23.89 | 17.99 | 18.73 | 18.63 | 18.73 | 17.76 |
| | | RMSE | **8.68** | 10.88 | 8.82 | 9.10 | 9.05 | 9.31 | 8.94 |
| METR-LA | 15min | MAE | 2.66 | 3.01 | **2.63** | 2.68 | 2.68 | 2.67 | 2.68 |
| | | MAPE | 7.06 | 8.03 | **6.98** | 7.10 | 7.11 | 7.09 | 7.13 |
| | | RMSE | **5.28** | 5.94 | 5.32 | 5.31 | 5.31 | 5.35 | 5.31 |
| | 30min | MAE | 3.02 | 3.27 | **3.01** | 3.06 | 3.06 | 3.06 | 3.04 |
| | | MAPE | 8.49 | 9.20 | **8.46** | 8.56 | 8.59 | 8.67 | 8.51 |
| | | RMSE | 6.28 | 6.68 | 6.36 | 6.32 | **6.27** | 6.36 | 6.32 |
| | 60min | MAE | **3.40** | 3.72 | 3.41 | 3.46 | 3.47 | 3.49 | 3.45 |
| | | MAPE | **9.88** | 10.94 | **9.88** | 10.02 | 10.07 | 10.34 | 9.87 |
| | | RMSE | **7.24** | 7.70 | 7.28 | 7.31 | 7.25 | 7.39 | 7.32 |
| | 90min | MAE | **3.64** | 4.09 | 3.65 | 3.71 | 3.73 | 3.75 | 3.69 |
| | | MAPE | **10.74** | 12.25 | 10.85 | 10.87 | 10.98 | 11.30 | 10.63 |
| | | RMSE | 7.74 | 8.38 | **7.71** | 7.81 | 7.75 | 7.91 | 7.81 |

- SimpleMem : Memory slot의 단순 pattern attention을 적용했을 때 (도로 간 거리 관계 무시)
  - 성능 감소가 나타남 → Graph Structure를 고려한 attention을 함께 사용하는 것이 유의미함
  - Graph Structure Attention만을 고려하는 GCN 모델에 비해서도 성능이 낮아짐을 볼 수 있음

# Experiments

## Evaluation

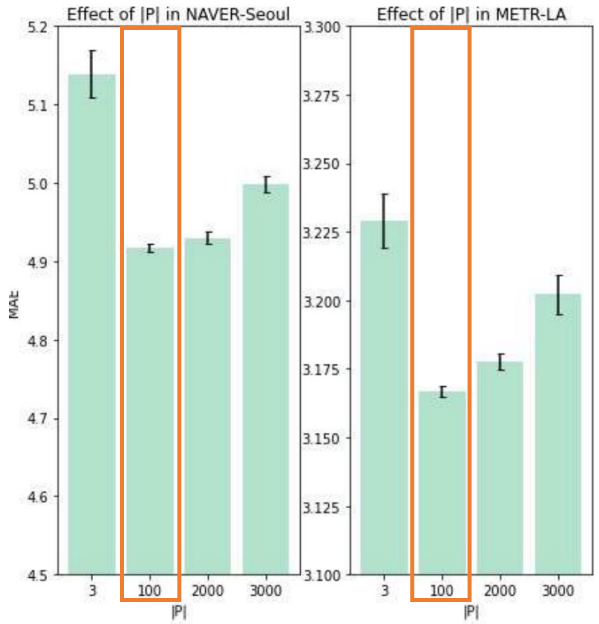


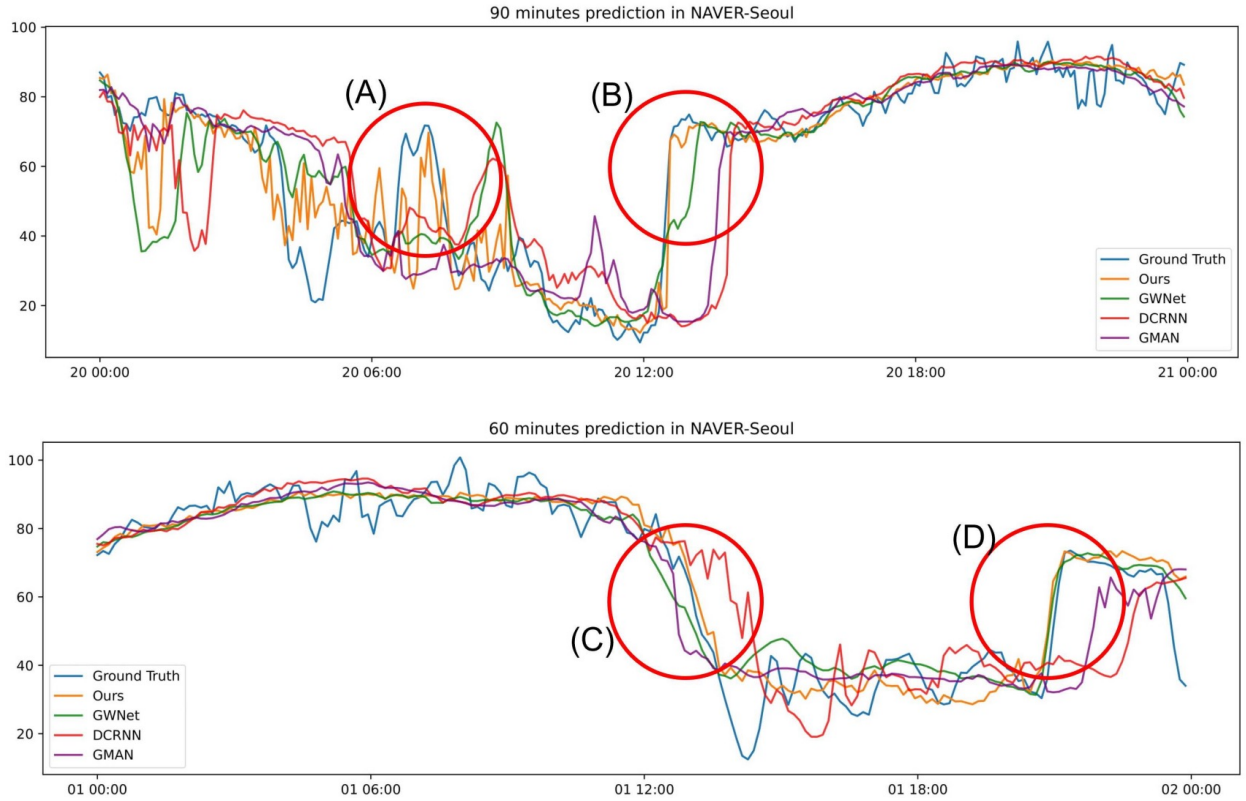

Table 2: Ablation study result. Note that 'Ours' means PM-MemNet.

| Ours | SimpleMem | CNN Decoder | RNN Decoder | Ours (L=1) | Ours ($\|\mathbb{P}\| = k$) | Ours ($\|\mathbb{P}\| >> 1000$) |
|---|---|---|---|---|---|---|
| 4.57 | 5.72 | **4.56** | 4.67 | 4.72 | 4.66 | 4.59 |
| 14.43 | 18.18 | **14.40** | 14.84 | 14.98 | 14.77 | 14.48 |
| 6.72 | 8.79 | **6.71** | 6.83 | 6.87 | 6.89 | 6.73 |
| **5.04** | 5.84 | 5.06 | 5.19 | 5.22 | 5.21 | 5.09 |
| **16.34** | 18.86 | 16.36 | 16.87 | 16.97 | 16.91 | 16.41 |
| **7.86** | 9.24 | 7.90 | 8.02 | 8.03 | 8.18 | 7.97 |
| **5.24** | 6.38 | 5.32 | 5.47 | 5.52 | 5.53 | 5.36 |
| **16.94** | 21.42 | 17.19 | 17.91 | 17.97 | 18.05 | 17.19 |
| **8.39** | 10.08 | 8.51 | 8.69 | 8.70 | 8.92 | 8.67 |
| **5.40** | 6.95 | 5.55 | 5.70 | 5.72 | 5.74 | 5.52 |
| **17.44** | 23.89 | 17.99 | 18.73 | 18.63 | 18.73 | 17.76 |
| **8.68** | 10.88 | 8.82 | 9.10 | 9.05 | 9.31 | 8.94 |
| 2.66 | 3.01 | **2.63** | 2.68 | 2.68 | 2.67 | 2.68 |
| 7.06 | 8.03 | **6.98** | 7.10 | 7.11 | 7.09 | 7.13 |
| **5.28** | 5.94 | 5.32 | 5.31 | 5.31 | 5.35 | 5.31 |
| 3.02 | 3.27 | **3.01** | 3.06 | 3.06 | 3.06 | 3.04 |
| 8.49 | 9.20 | **8.46** | 8.56 | 8.59 | 8.67 | 8.51 |
| 6.28 | 6.68 | 6.36 | 6.32 | **6.27** | 6.36 | 6.32 |
| **3.40** | 3.72 | 3.41 | 3.46 | 3.47 | 3.49 | 3.45 |
| **9.88** | 10.94 | **9.88** | 10.02 | 10.07 | 10.34 | 9.87 |
| **7.24** | 7.70 | 7.28 | 7.31 | 7.25 | 7.39 | 7.32 |
| **3.64** | 4.09 | 3.65 | 3.71 | 3.73 | 3.75 | 3.69 |
| **10.74** | 12.25 | 10.85 | 10.87 | 10.98 | 11.30 | 10.63 |
| 7.74 | 8.38 | **7.71** | 7.81 | 7.75 | 7.91 | 7.81 |

- 모델이 경량화 된 상태(Layer = 1)에서도 기존 모델 대비 좋은 성능을 확인할 수 있음
- Pattern 추출을 3개, 1000개 이상 뽑은 Case
  - $|\mathbb{P}| = [2000, 3000]$ 을 사용해 볼 때, 오히려 성능의 저하를 야기하는 경우 발생
  - 메모리 슬롯 자체가 유사성 위주로 진행되기 때문에 패턴 슬롯 크기를 키우면 오히려 Overfitting 문제

# Experiments

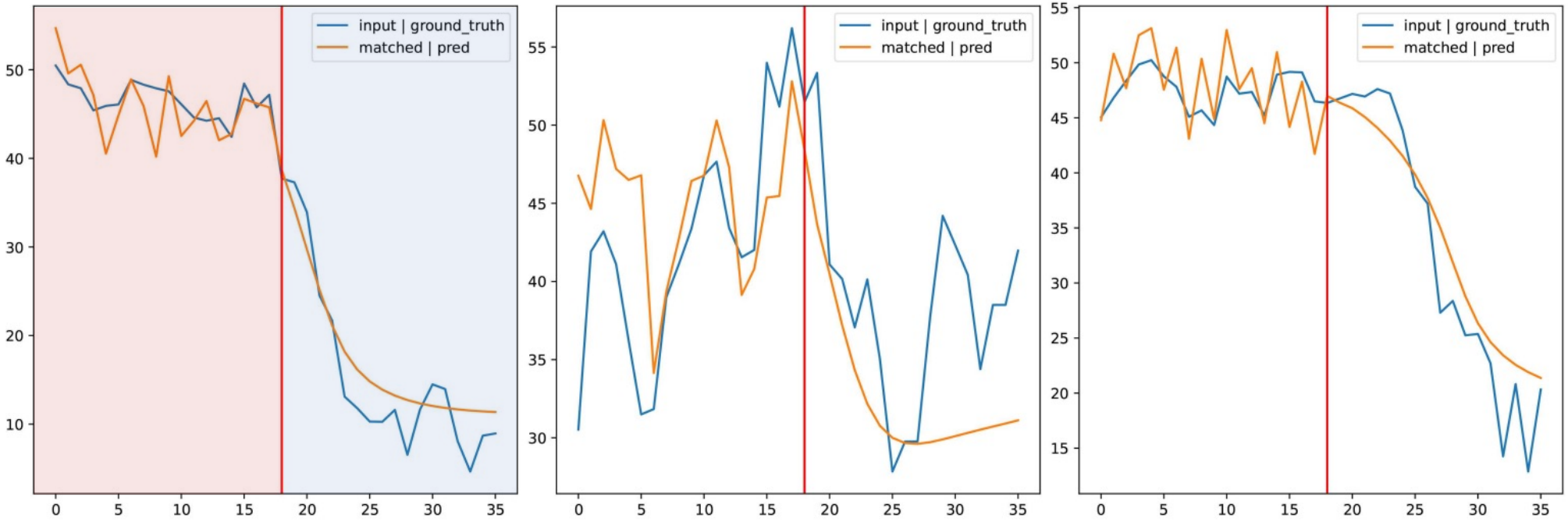
## Evaluation




- 타 모델 대비, 급격한 변화 발생 시 변화를 따라잡는 시간이 상대적으로 빠름

# Experiments

## Evaluation




- Input이 들어오면 Memory Slot에 유사한 패턴이 Matching이 되고 있음
- 갑작스런 Speed Drop 상황에서 이뤄지는 예측이 GT의 추세를 따라가고 있는 것을 확인할 수 있음

# Conclusion

## Limits

- Memory Pattern 추출하기 위한 사전 전처리
  - 최적의 Hyper Parameter 값을 실험을 통해 찾아야 함
- Memory Slot의 update가 자주 나오는 패턴에 의존적임
  - 거의 나오지 않는 패턴은 무시되는 경향이 존재
  - 한번도 보지 못한 패턴에 대해서는 정확하지 않을 가능성이 존재함
- Cosine Similarity를 통한 Pattern Matching
  - Traffic Data는 특성 상 Noise가 다수 존재함
  - Cosine Similarity를 통한 Clustering 방법은 완전하다고 볼 수 없음

# Conclusion

## Contributions

- GCN + Learnable Adjacency Matrix + Mem2Seq 융합

- Traffic Flow Forecasting에 대해 Pattern Matching으로의 사고 전환
  - 더 많은 정보를 가진 input을 넣는 대신 대표하는 패턴을 input으로 robustness 확보
    - 실제 교통 데이터는 통제 불가능한 Noise가 다수 존재
    - 시계열 자체에 temporal attention을 사용 → 시계열 특징 저해 가능성
  - Noise를 robust하게 해결하는 Task에 다방면으로 적용 가능할 것이라고 생각됨

감사합니다