Transactions of KSAE, Vol. 34, No. 3, pp.365-373 (March, 2026)

pISSN 1225-6382 / eISSN 2234-0149
DOI http://dx.doi.org/10.7467/KSAE.2026.34.3.365

효율성과 견고성을 갖춘 궤적 예측을 위한 상호작용 그래프 가지치기

무하마드 라만[1,2] · 홍 정 빈[1,2] · 최 두 섭[*1,2] · 민 경 욱[1]

한국전자통신연구원 자율주행지능연구실[1] · 과학기술연합대학원대학교 인공지능학과[2]

Interaction Graph Pruning for Efficient and Robust Trajectory Prediction

Muhammad Atta Ur Rahman[1,2] · Jeongbin Hong[1,2] · Dooseop Choi[*1,2] · Kyoung Wook Min[1]

[1]Autonomous Driving Intelligence Research Section, ETRI, 218 Gajeong-ro, Yusung-gu, Daejeon 34129, Korea
[2]Department of Artificial Intelligence, University of Science and Technology, 217 Gajeong-ro, Yusung-gu, Daejeon 34113, Korea
(Received 3 September 2025 / Revised 14 October 2025 / Accepted 14 October 2025)

Abstract : Modeling interactions between dynamic and static road participants(e.g., vehicles, pedestrians, lane dividers, road markings) is crucial for accurate and robust trajectory prediction. Transformer-based approaches with self- and cross-attention have become the standard due to their ability to capture complex, long-range dependencies. However, as the number of participants grows, complexity increases quadratically, imposing a significant burden during training and inference. To address this, the researchers propose IGP-Traj, an interaction graph pruning algorithm that eliminates dispensable connections from the fully connected interaction graph. IGP-Traj dynamically prunes irrelevant agent-to-agent(A2A) and agent-to-map(A2M) connections, focusing on critical interactions, such as nearest and forward-looking ones. This reduces GPU memory usage, accelerates training and inference, and enhances model performance. We evaluate IGP-Traj on three state-of-the-art Transformer-based models using the large-scale Argoverse 1 and 2 datasets, demonstrating its effectiveness in achieving efficient and robust trajectory prediction.

Key words : Autonomous driving(자율 주행), Artificial intelligence(인공 지능), Trajectory prediction(궤적 예측), Transformer(트랜스포머), Interaction graph(상호작용 그래프)

Nomenclature

N : number of interaction agents in a scene
T : number of timesteps
x_i^t : state vector of agent i at time t
p_i^t : position of agent i at time t, meters
θ_i^t : heading angle of agent i at time t, radians
v_i^t : speed of agent i at time t, m/s
g_l : l-th map segment in HD map
a_i^t : agent embedding vector of agent i at time t
m_l : map embedding vector of l-th map segment in HD map
q_k : k-th query vector
A_i^t : agent-to-agent interaction graph for agent i at time t
M_i^t : agent-to-map interaction graph for agent i at time t
$a_{j\to i}^t$: edge from neighbor j to target i in A2A graph
$m_{l\to i}^t$: edge from map segment l to target i in A2M graph
$s_{j\to i}^t$: importance score for A2A edge $j \to i$
$s_{l\to i}^t$: importance score for A2M edge $l \to i$
d_{ij}^t, d_{il}^t : Euclidean distances between agent j (or map segment l) and agent i
$\theta_{ij}^t, \theta_{il}^t$: relative heading angles between agent j (or map segment l) and agent i
τ_{il}^t, τ_{ij}^t : time-to-reach (TTR) between agent j (or map segment l) and agent i

*Corresponding author, E-mail: d1024.choi@etri.re.kr

1. 서 론

교통 에이전트(Traffic agents)의 미래 움직임을 예측하는 것은 자율주행에서 핵심 과제이며, 정확한 예측은 자율주행차(Autonomous Vehicle, AV)가 안전하게 주행할 수 있도록 한다.[1-3] 여기서 에이전트란 AV와 주변의 다른 차량, 보행자 등을 의미한다. 그동안 관련 연구가 활발히 이루어져 왔으며, 예측 과제의 주요 도전 과제 중 하나는 교통 에이전트와 도로 요소(차선 분리대, 노면 표시 등) 간 상호작용을 모델링하는 것이다.

주행 장면(Driving scene) 에서의 상호작용은 크게 에이전트 간(Agent-to-Agent, A2A)와 에이전트 및 지도(Agent-to-Map, A2M) 간 상호작용으로 나눌 수 있다. A2A 상호작용을 정확히 포착하는 것은 사회적 규범을 고려하면서 충돌 없는 미래 궤적 예측 시스템을 개발하는 데 필수적이다. A2M 상호작용 모델링은 교통 에이전트가 고정된 환경 요소(차선 중심선, 노면 표시 등)를 어떻게 인지하고 반응하는지를 이해하는 것을 의미한다. 이는 고정밀 지도(HD map)에 정의된 규칙에 따라 에이전트들이 도로 위를 움직이기 때문에 궤적 예측의 정확도에 중요한 역할을 한다. 이러한 상호작용을 데이터로부터 학습하려는 연구가 다수 제안되었으며, 최근에는 확장성과 장기 의존성을 포착하는 능력 덕분에 Transformer[4] 기반 접근법이 주류가 되었다.

그러나 주행 장면의 각 에이전트가 반드시 모든 에이전트나 도로 요소와 상호작용하는 것은 아니다. Transformer의 Self- 및 Cross-attention 메커니즘은 불필요한 상호작용을 자동으로 구분하여 성능을 향상시키지만, Attention 연산의 복잡도가 $O(N^2)$ 로 증가해 연산 비용이 크다는 문제가 있다. 따라서 특정 에이전트의 미래 움직임을 예측할 때는 해당 에이전트와 관련 없는 이웃 에이전트나 도로 요소를 배제하는 것이 합리적이며, 이를 다루려는 연구들이 일부 제안되었다.[5-7] 하지만 기존 방법들은 A2M 상호작용을 충분히 반영하지 못해 최적의 성능을 내지 못하고, 상호작용 판별을 위해 추가적인 신경망과 연산을 사용하기에 모델 복잡도와 추론 속도 저하를 초래한다.

본 논문에서는 Transformer 기반 차량 궤적 예측 모델을 위해 IGP-Traj라 명명한 새로운 상호작용 그래프 가지치기 알고리즘을 제안한다. 제안된 방법은 차량이 도로 위에서 움직이는 방식을 관찰한 결과를 반영하여, 에이전트와 도로 요소 간의 불필요한 상호작용을 식별할 수 있도록 정교하게 설계되었다. 이는 학습 기반 방법이 아니므로, 기존 접근법에서 흔히 발생하는 막대한 계산 비용이 필요하지 않다. 우리는 식별 과정을 위해 세 가지 마스크(전방, 도달 시간, 근접성 마스크)를 도입하는데, 각각은 특정 에이전트나 지도 요소가 목표 에이전트와 얼마나 연관성이 있는지를 측정한다. 매 시점마다 에이전트의 상태 정보와 지도 요소가 주어지면, 세 가지 마스크는 동적으로 계산되어 Attention 연산 과정에서 목표 에이전트와 무관한 요소를 배제하는 데 사용된다. 이로써 계산 복잡도와 GPU 메모리 사용량이 줄어들어 더 빠른 추론이 가능해지고 성능도 향상된다.

2. 관련 연구

2.1 에이전트 간 상호작용

A2A 모델링은 크게 세가지 방식으로 나뉜다. 먼저 Convolutional Neural Network(CNN) 기반의 방식은 2D 이미지 위에 자율차를 비롯한 주변 에이전트들의 과거 위치 기록을 2D bounding box의 형태로 그린 후, CNN을 통해 이미지를 피쳐맵(Feature map)의 형태로 변환한다.[8,9] 이 때 2D 이미지의 각 채널은 특정 시점에서 AV 또는 목표 에이전트에 대한 주변 에이전트들의 상대적 위치를 나타내므로, 시·공간적으로 A2A 상호작용을 효과적으로 포착할 수 있다.

Graph Neural Network(GNN) 기반의 방식은 각 에이전트를 노드(Node)로, A2A 상호작용을 엣지(Edge)로 표현함으로써 에이전트들 사이의 시·공간적 의존성을 효율적으로 포착한다.[10,11] 특히 에이전트들의 시간에 따른 가시성과 연관성에 기반하여 엣지가 갱신되는 동적 시·공간 그래프는 에이전트들 사이의 시간에 따른 상호작용 변화를 잘 포착하는 것으로 알려져 있다.

Transformer 기반의 방식은 각 에이전트를 학습이 가능한 토큰(Token) 또는 피쳐벡터(Feature vector)로 표현하고 Self- 및 Cross-attention 메커니즘을 통해 각 토큰을 갱신함으로써 A2A 상호작용을 모델링한다.[12-14] 토큰의 갱신 과정에서 계산되는 Attention score는 두 에이전트 사이의 연관성을 수치적으로 표현하며 그 값에 따라 한 에이전트가 다른 에이전트에 미치는 영향력을 조절한다. 따라서 Transformer 기반의 방식은 시·공간적으로 매우 멀리 떨어진 에이전트들 간의 상호작용도 데이터를 통해 효과적으로 학습할 수 있다는 장점이 있다. 이러한 이유로 근래의 대부분의 연구는 Transformer를 기반으로 수행되고 있다고 해도 과언이 아니다.

2.2 에이전트 및 지도간 상호작용

A2M 상호작용 모델링 또한 A2A 모델링과 동일하게 CNN 기반, GNN 기반, 그리고 Transformer 기반 방식으로 구분할 수 있다. 먼저 CNN 기반의 방식은, 앞선 A2A 모델링 방식과 유사하게, 차로 중심선과 같은 정밀 지도 성

분을 2D 이미지 위에 그린 후 CNN을 통해 피쳐맵으로 변환한다.[8,9] 이 때 이미지의 각 채널이 서로 다른 지도 성분을 나타내도록 함으로써 피쳐맵을 통해 도로 규칙을 보다 정밀하게 포착할 수 있다.

다음으로 GNN기반 방식은 에이전트 및 정밀 지도의 각 세그먼트(Segment)를 노드로 표현하고 에이전트와 세그먼트 사이의 상호작용을 엣지로 표현한다.[1] 하나의 그래프 내에 에이전트 및 지도 세그먼트 노드와 엣지가 공존하도록 설계함으로써 학습 과정에서 A2A 및 A2M을 동시에 모델링이 가능하도록 할 수 있으며 반대로 A2A 및 A2M을 위한 별도의 그래프를 설계하여 두 상호작용을 독립적으로 모델링 할 수도 있다.

마지막으로 Transformer 기반의 방식은 정밀 지도의 각 세그먼트를 토큰으로 표현하고 Self- 및 Cross-attention 메커니즘을 통해 각 토큰을 갱신함으로써 A2M 상호작용을 모델링한다.[9-11] GNN 방식과 마찬가지로 에이전트 및 지도 세그먼트 토큰이 하나의 Transformer 모델 내에서 상호작용하도록 설계하거나 혹은 A2A 및 A2M을 위한 별도의 Transformer를 설계하여 두 상호작용을 독립적으로 모델링 할 수도 있다.

2.3 상호작용 그래프 가지치기

RAIN [6]은 강화학습(Reinforcement Learning, RL) 기반의 Hard-attention 방법을 제안하였으며, 여기서 RL 네트워크는 목표 에이전트와 가장 관련이 깊은 주변 에이전트만을 선별하도록 학습된다. 이렇게 선별된 주변 에이전트들은 이후 Graph message passing을 위해 Soft-attention 연산을 통해 중요도 점수가 부여된다. GP-Graph[5]는 같은 이동 그룹에 속하지 않는 에이전트의 엣지를 마스킹(Masking)하여 각 에이전트를 가장 가능성 높은 행동 그룹에 할당하도록 학습하며, 이를 통해 그룹 내 공통적인 움직임을 더 잘 모델링한다. FJMP[7]는 선별 문제를 분류 과제로 보고, 상호작용 그래프 내 각 방향성 엣지를 Interaction 혹은 No-interaction으로 분류하는 전용 신경망을 학습한다. 기존의 학습 기반 가지치기 방식은 가지치기를 위한 별도의 네트워크를 모델 내에 포함하므로, 필연적으로 학습 및 추론 시 더 많은 GPU 메모리와 연산을 요구한다.

3. 제안하는 방법

3.1 문제의 정의

N개의 상호작용하는 에이전트가 있는 교통 상황을 가정하자. 우리의 목표는 각 에이전트에 대해 미래 궤적 분포 $p(\boldsymbol{Y_i}|\{\boldsymbol{X}_i\}_{i=1}^{N},\{\boldsymbol{g}_l\}_{l=1}^{M})$ 를 생성하는 것이다. 여기서 $\boldsymbol{X}_i \in R^{T_h \times d}$ 와 $\boldsymbol{Y}_i \in R^{T_f \times 2}$는 각각 에이전트 V_i의 과거 상태 이력과 미래 궤적을 나타낸다. T_h와 T_f는 각각 과거와 미래의 시간 지평(Time horizon)을 의미하며, d는 에이전트 상태 벡터의 차원이다. 본 논문에서는 시간 t에서의 상태 벡터 $\boldsymbol{x}_i^t \in R^d$에 대해, 에이전트의 위치 좌표 $\boldsymbol{p}_i^t \in R^2$, 진행 방향 각도 $\theta_i^t \in R^1$, 속도 $v_i^t \in R^1$를 고려한다. $\boldsymbol{g}_l \in R^{L \times d}$은 현재 주행 장면에서 사용 가능한 정밀 지도 세그먼트를 나타내며, 차선 구간 상의 균등한 간격의 좌표점들, 각 위치점에서의 방향, 그리고 그 유형 등으로 표현된다.

3.2 일반적인 Transformer 기반의 네트워크 구조

Transformer 기반 궤적 예측 모델은 일반적으로 다음과 같은 데이터 처리 구조를 공유한다. 인코더(Encoder) 측에서는, 먼저 Multi-Layer Perceptron(MLP)를 이용해 에이전트의 과거 상태 및 맵 세그먼트로부터 에이전트 임베딩(Embedding) $\boldsymbol{a}_i^t \in R^D$과 맵 임베딩 $\boldsymbol{m}_l \in R^D$을 생성한다. 이렇게 생성된 임베딩들은 이후 Factorized Attention (FA) 또는 Multi-Axis Attention(MA)을 거쳐 에이전트와 맵 요소 간의 상호작용을 반영하여 스스로를 업데이트한다. 디코더(Decoder) 측에서는 학습 가능하고 무작위로 초기화된 다수의 쿼리 벡터(Query vector) $\{\boldsymbol{q}_k \in R^D\}$가 업데이트된 임베딩들에 Attention을 수행하고 이에 따라 자신을 갱신한 후, MLP를 거쳐 다중 모달 궤적을 생성한다. Fig. 1은 이러한 Transformer 기반 궤적 예측 모델의 구조를 가시화하고 있다.

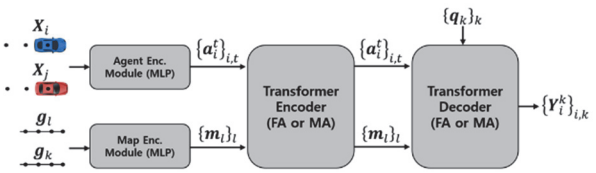

Fig. 1 Overview of a Transformer-based prediction model architecture

Multi-axis Attention(MA)는 공간 차원과 시간 차원 모두에서 Self-attention을 동시에 수행하는 기본적인 Transformer 설정을 의미한다. 구조적으로 단순하지만 계산 비용이 크며, 그 복잡도는 $O(N^2 \times T^2)$로 나타난다. 반면, Factorized Attention(FA)는 공간과 시간에 대해 순차적으로 Self- 또는 Cross-attention을 적용하여 이러한 복잡도 문제를 완화한다. 이 경우 전체 복잡도는 $O(N^2) + O(T^2)$로 줄어들며, 그럼에도 불구하고 성능은 MA와 유사한 수준을 달성한다.[15] 따라서, 본 논문에서는 FA를 가정하여 이후 설명을 진행한다. 그러나 제안하는 방법은 MA를 사용하는 모델에도 쉽게 적용 가능함을 미리 밝힌다.

3.3 제안하는 알고리즘

Fig. 2는 IGP-Traj 알고리즘을 통합한 FA 기반 예측 모델의 인코더 아키텍처를 간략히 보여준다. 매 시각 t에서, 에이전트 V_i는 두 개의 상호작용 그래프 $\boldsymbol{A}_i^t$와 $\boldsymbol{M}_i^t$를 가지며, 전자는 A2A 연결을, 후자는 에이전트와 맵 세그먼트 간 연결을 나타낸다. $a_{j\to i}^t \in \{0,1\}$과 $m_{l\to i}^t \in \{0,1\}$을 각각 그래프 $\boldsymbol{A}_i^t$와 $\boldsymbol{M}_i^t$의 엣지라 하자. 여기서 $j \to i$ 는 V_j가 V_i에 미치는 영향을, $l \to i$은 $\boldsymbol{g}_l$이 V_i에 미치는 영향을 의미한다. 초기에는 두 그래프의 모든 엣지들이 1로 설정된다. IGP 모듈은 매 시각 t마다 중요하지 않은 엣지들을 식별하고 해당 값을 0으로 설정함으로써 두 그래프를 갱신한다. 이 후 두 그래프는 A2A 및 A2M 상호작용 모델링 과정에서 에이전트가 엣지 값이 1인 에이전트와 맵 세그먼트에 주의를 기울이도록 강제한다. Fig. 3은 그 예를 가시화하고 있다. Fig. 2의 아키텍처는 예측 모델의 인코더를 염두에 두고 설명된 것이지만 디코더로 쉽게 일반화할 수 있다. 예를 들어, 미래 예측 시점 $t+T$에서 예측을 위해 업데이트 된 에이전트 임베딩들은 다음 예측 시점 $t+T+1$에서 FA 모듈을 통해 다시 업데이트된다. 이 과정에서 IGP 모듈이 개입되며, 쿼리들은 업데이트 된 임베딩들과의 Attention을 통해 새롭게 업데이트 되고 시점 $t+T+1$에서의 예측에 활용된다.

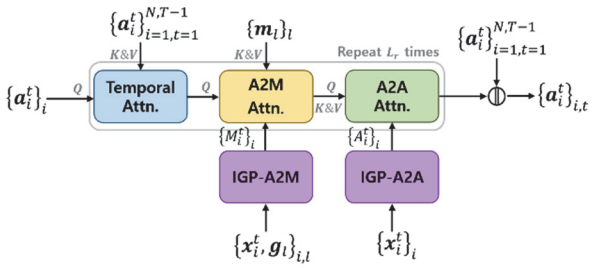

Fig. 2 Overview of an encoder architecture of a FA-based prediction model incorporating the proposed IGP algorithm. At each time $\boldsymbol{t}$, agent embeddings $\{\boldsymbol{a}_i^t\}$ go through Temporal, A2M, and A2A attention blocks to update themselves. IGP-A2M and IGP-A2A respectively generate two interaction graphs $\boldsymbol{A}_i^t$ and$\boldsymbol{M}_i^t$, each stopping agent embeddings from being updated by irrelevant information

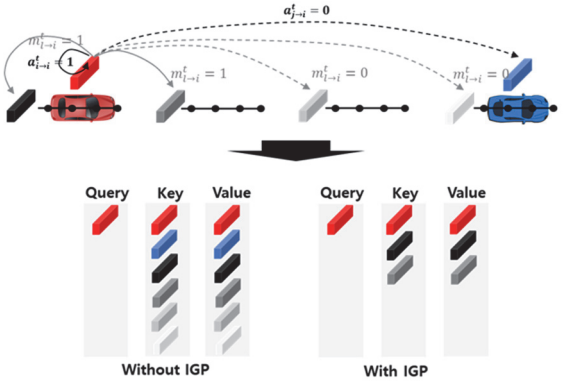

Fig. 3 Example of self-attention operation in Transformer with the proposed IGP method

마지막으로 Child et al.[16]은 Transformer의 soft-attention 연산 과정 시 쿼리가 참조하는 데이터의 범위를 제한하여 연산 속도를 높이는 Sparse attention 기법을 제안하였다. 본 논문의 IGP-Traj 알고리즘은 Sparse attention 기법의 특별 케이스로, 목표 차량이 참조하는 주변 차량 또는 맵 세그먼트를 앞으로 설명할 기준에 따라 제한하여 Transformer의 메모리 사용량을 감소하고 연산 속도를 높인다.

3.3.1 IGP-A2M module

앞서 말한 바와 같이 그래프 $\boldsymbol{M}_i^t$의 엣지 $m_{l\to i}^t$는 초기에 1의 값을 갖는다. IGP-A2M 모듈은 $m_{l\to i}^t$을 아래와 같이 중요도 스코어 $s_{l\to i}^t \in R^1$을 통하여 업데이트 한다.

$$m_{l\to i}^t = \begin{cases} 1, & s_{l\to i}^t > Thr \\ 0, & otherwise \end{cases} \tag{1}$$

여기서 Thr는 실험을 통해 정하는 역치 값이다. 중요도 스코어 $s_{l\to i}^t$는 아래와 같이 세가지 측정값들$(d_{il}^t, \theta_{il}^t, \tau_{il}^t)$의 가중치 합으로 결정된다.

$$s_{l\to i}^t = -d_{il}^t + \mu_1^m \cdot \left[\theta_{il}^t < \frac{\pi}{2}\right] + \mu_2^m \cdot exp[-\rho \cdot \tau_{il}^t] \tag{2}$$

여기서 μ 와 ρ 는 실험을 통해 결정하는 상수이며, $[condition]$은 조건이 성립할 때 1의 값을 반환하는 함수이다. 본 논문에서는 $\rho = 0.05$로 μ는 실험을 통해 찾았다. 세가지 측정값 들은 아래와 같이 정의된다.

● d_{il}^t : 시간 t에서 $\boldsymbol{g}_l$와 V_i사이의 유클리드 거리이다. 이 측정값은 운전자가 주로 주변 차선 정보를 참고하여 주행함을 바탕으로 도출되었다. 수식 (2)에 따라, 운전자로부터 거리가 먼 맵 세그먼트는 낮은 중요도 스코어를 유도한다.

● θ_{il}^t : 시간 t에서 V_i가 $\boldsymbol{g}_l$을 마주하기 위해 회전해야 할 각도이 이며 다음과 같이 계산한다.

$$\theta_{il}^t = arccos\left(\frac{\boldsymbol{p}_i^t - \boldsymbol{p}_i^{t-1}}{||\boldsymbol{p}_i^t - \boldsymbol{p}_i^{t-1}||_2} \circ \frac{\boldsymbol{p}_{g_l} - \boldsymbol{p}_i^t}{||\boldsymbol{p}_{g_l} - \boldsymbol{p}_i^t||_2}\right) \tag{3}$$

여기서 $\circ$은 내적(Inner product)를 의미한다. 식 (3)의 내적 기호 왼쪽은 차량의 헤딩 벡터를, 오른쪽은 세그먼트 $\boldsymbol{g}_l$

의 중심점과 차량의 현재 위치를 이은 벡터를 의미한다. 이 측정값은 운전자가 주로 전방을 주시하며 운전함을 바탕으로 도출되었다. 수식 (2)에 따라, 차량의 전방에 있는 맵 세그먼트만이 중요도 스코어에 영향을 미친다.

● τ_{il}^t : 시간 t에서 V_i가 $\boldsymbol{g}_l$에 도달하기까지 걸리는 시간을 의미하며 다음과 같이 계산된다.

$$\tau_{il}^t = \begin{cases} \frac{d_{il}^t}{v_{proj}^t}, & v_{proj}^t > 0 \\ \infty, & otherwise \end{cases} \tag{4}$$

여기서 v_{proj}^t는 V_i의 $\boldsymbol{g}_l$방향으로의 속도이며 아래와 같이 계산된다.

$$v_{proj}^t = (\boldsymbol{p}_i^t - \boldsymbol{p}_i^{t-1}) \circ \frac{\boldsymbol{p}_{g_l} - \boldsymbol{p}_i^t}{||\boldsymbol{p}_{g_l} - \boldsymbol{p}_i^t||_2} \tag{5}$$

이 측정값은 운전자가 속도에 따라 가깝거나 혹은 멀리 있는 차선에 다르게 집중함을 반영하기 위해 제안되었다. 차량의 속도가 빠른 경우, 비록 멀리 있는 차선이라 하더라도 더욱 주시하여 운전함이 그 예이다. 참고로 $v_{proj}^t \leq 0$인 경우, 차량이 맵 세그먼트를 뒤에 두고 멀어져가는 상황이므로 $\tau_{li}^t = \infty$로 두어 해당 맵 세그먼트의 중요도 스코어를 높이지 않도록 하였다.

Fig. 4는 각 측정값이 자율차와 맵 세그먼트의 관계에 따라 어떻게 변하는지 시각화 하여 보여주고 있다. 먼저 그림 안의 검은색 박스는 자율차의 현재 위치를 보여준다. 다음으로 맵 세그먼트, 즉 차로 중심선에 부여된 측정값의 크기에 따라 색을 다르게 표시하였다. 노란색에 가까울수록 높은 값을 그리고 파란색에 가까울수록 작은 값을 의미한다. 마지막으로 검은색은 수식 (4)에서 무한대인 경우를 의미한다. Fig. 4(a)는 차량과 세그먼트 사이의 거리에 따라 값이 변하는 모습을, (b)는 차량의 전방에 해당되는 세그먼트만이 선택된 모습을, 마지막으로 (c)는 차량이 빨리 도달하게 되는 세그먼트만이 선택된 모습을 볼 수 있다.

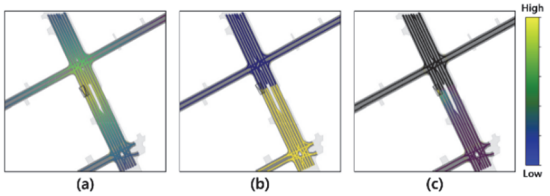

Fig. 4 Visualization of (a) $-d_{il}^t$, (b) $[\theta_{il}^t < \pi/2]$, and (c) $exp[-\rho \cdot \tau_{il}^t]$ for the importance score $s_{l\to i}^t$ in IGP-A2M module. The values are color-coded with yellow and blue where yellow indicates the highest value and blue indicates the lowest one. The black in (c) indicates the infinity. Please zoom in for better visibility

3.3.2 IGP-A2A module

그래프 $\boldsymbol{A}_i^t$의 엣지 $a_{j\to i}^t$는 초기에 1의 값을 갖는다. IGP-A2M와 유사한 방식으로, IGP-A2A 모듈은 $a_{j\to i}^t$을 아래와 같이 중요도 스코어 $s_{j\to i}^t \in R^1$를 통하여 업데이트 한다.

$$a_{j\to i}^t = \begin{cases} 1, & s_{j\to i}^t > Thr \\ 0, & otherwise \end{cases} \tag{6}$$

여기서 Thr는 실험을 통해 정하는 역치 값이다. 중요도 스코어 $s_{j\to i}^t$는 아래와 같이 세가지 측정값들$(d_{ij}^t, \theta_{ij}^t, \tau_{ij}^t)$의 가중치 합으로 결정된다.

$$s_{j\to i}^t = -d_{ij}^t + \mu_1^a \cdot \left[\theta_{ij}^t < \frac{\pi}{2}\right] + \mu_1^a \cdot exp[-\rho \cdot \tau_{ij}^t] \tag{7}$$

본 논문에서는 $\rho = 0.05$로 μ는 실험을 통해 찾았다. 세 가지 측정값 들은 아래와 같이 정의된다.

● d_{ij}^t : 시간 t에서 V_j와 V_i사이의 유클리드 거리이다. 이 측정값은 운전자가 주로 주변 차량 정보를 참고하여 주행함을 바탕으로 도출되었다. 수식 (7)에 따라, 운전자로부터 거리가 먼 차량은 낮은 중요도 스코어를 유도한다.

● θ_{ij}^t : 시간 t에서 V_i가 V_j을 마주하기 위해 회전해야 할 각도이 이며 다음과 같이 계산한다.

$$\theta_{ij}^t = arccos\left(\frac{\boldsymbol{p}_i^t - \boldsymbol{p}_i^{t-1}}{||\boldsymbol{p}_i^t - \boldsymbol{p}_i^{t-1}||_2} \circ \frac{\boldsymbol{p}_j^t - \boldsymbol{p}_i^t}{||\boldsymbol{p}_j^t - \boldsymbol{p}_i^t||_2}\right) \tag{8}$$

식 (8)의 내적기호 왼쪽은 차량 V_i의 헤딩 벡터를, 오른쪽은 V_j와 V_i의 현재 위치를 이은 벡터를 의미한다. 이 측정값은 운전자가 주로 전방을 주시하며 운전함을 바탕으로 도출되었다. 수식 (7)에 따라, 목표 차량의 전방에 있는 에이전트만이 중요도 스코어에 영향을 미친다.

● τ_{ij}^t : 시간 t에서 V_j가 V_i에 도달하기까지 걸리는 시간을 의미하며 다음과 같이 계산된다.

$$\tau_{ij}^t = \begin{cases} \frac{d_{ij}^t}{v_{proj}^t}, & v_{proj}^t > 0 \\ \infty, & otherwise \end{cases} \tag{9}$$

여기서 v_{proj}^t는 V_j의 V_i방향으로의 속도이며 아래와 같이 계산된다.

$$v_{proj}^t = (\boldsymbol{p}_j^t - \boldsymbol{p}_j^{t-1}) \circ \frac{\boldsymbol{p}_i^t - \boldsymbol{p}_j^t}{||\boldsymbol{p}_i^t - \boldsymbol{p}_j^t||_2} \tag{10}$$

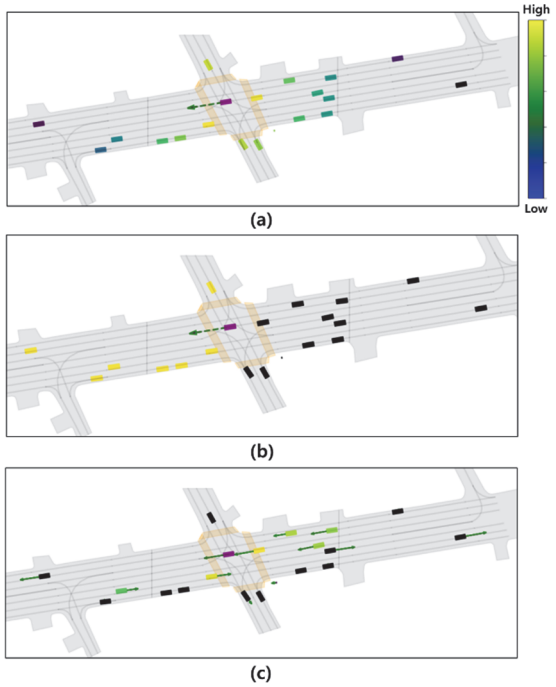

Fig. 5 Visualization of (a) $-d_{ij}^{t}$, (b) $\left[\theta_{ij}^{t} < \pi/2\right]$, and (c) $exp\left[-\rho \cdot \tau_{ij}^{t}\right]$ for the importance score $s_{j\to i}^{t}$ in IGP-A2A module. The values are color-coded with yellow and blue where yellow indicates the highest value and blue indicates the lowest. The black in (c) indicates the infinity. Please zoom in for better visibility

주변 차량 V_j가 목표 차량 V_i의 뒤에 위치하여 θ_{ij}^{t}에 의해 제외되더라도, τ_{ij}^{t}는 V_j가 빠른 속도로 V_i에 접근하는 경우 V_j에 의한 중요도 스코어 변화를 허용한다.

Fig. 5는 각 측정값이 자율차(V_i)와 주변 차량(V_j)의 관계에 따라 어떻게 변하는지 보여주고 있다. 먼저 그림의 보라색 박스는 자율차의 현재 위치를 보여준다. 다음으로 주변 차량에 부여된 측정 값에 따라 색을 다르게 표시하였다. 노란색에 가까울수록 높은 값을 그리고 파란색에 가까울수록 작은 값을 의미한다. 마지막으로 검은색은 수식 (9)에서 무한대인 경우를 의미한다. Fig. 5(a)는 차량 사이의 거리에 따라 값이 변하는 모습을, (b)는 자율차의 전방에 있는 차량만이 선택된 모습을, 마지막으로 (c)는 자율차에 빨리 도달하게 되는 주변 차량만이 선택된 모습을 볼 수 있다.

4. 실 험

4.1 데이터 셋

본 논문에서는 대규모 실제 데이터 셋 Argoverse 1[2]과 Argoverse 2[17]를 이용하여 제안하는 방식을 평가한다. 이 데이터셋은 훈련 및 검증용으로 최대 25만 개의 주행 시나리오를 포함하며, 각 시나리오의 길이는 5초(Argoverse 1, 처음 2초는 관찰, 이후 3초는 예측에 사용) 및 11초(Argoverse 2, 처음 5초는 관찰, 이후 6초는 예측에 사용)이다. 10 Hz로 샘플링 되었으며, 미국 6개 도시에서 수집되었다. 모든 시나리오는 에이전트들의 상태정보 뿐 아니라 연관된 정밀 지도 요소들도 제공하여, 예측 모델이 에이전트들과 환경 간의 내재적 상호작용을 데이터로부터 학습하고 포착할 수 있도록 한다.

4.2 Baseline 모델 및 구현

제안된 알고리즘의 효과를 입증하기 위해 세 가지 Transformer 기반 최신 예측 모델인 QCNet,[13] HiVT,[14] 그리고 HPNet[15]을 선정하였다. 3절에서 설명한 바와 같이, IGP-A2A 및 IGP-A2M 모듈은 궤적 인코딩 및 디코딩 과정의 각 시점 t에서 해당 모델들에 적용된다. 참고로, IGP-Traj 구현 시 목표 에이전트 근방 15 m 이내의 에이전트 및 맵 세그먼트는 항상 엣지 값이 1이 되도록 강제하였다. 그 이유는 애초에 주변 에이전트 및 세그먼트의 개수가 매우 적을 경우, 강제로 상위 70 %만을 상호작용을 위해 선택하도록 하는 것이 성능을 하락시키는 요인이 되기 때문이다.

제안하는 알고리즘과의 비교를 위해 기존의 학습 기반 그래프 가지치기 방법인 GP-Graph[5]를 세 모델에 적용하여 성능을 비교하였다. GP-Graph는 기본적으로 $a_{j\to i}^{t}$을 데이터로부터 학습하도록 제안된 모델이나 본 논문에서는 $m_{l\to i}^{t}$로도 확장하여 적용하였다. 참고로 공정한 평가를 위해 각 저자가 제공한 공식 구현 코드를 사용하였음을 밝힌다. 또한 모델을 학습할 때 공개된 코드에서 제시된 네트워크 구조와 학습 파라미터 설정을 엄격히 따랐다. 제안하는 IGP-Traj의 파라미터(예: μ, ρ)는 실험적으로 결정되었으며, 이후 절에서 모델의 성능이 이에 따라 어떻게 달라지는지를 보일 것이다. 각 방법들을 객관적으로 비교하기 위해, Argoverse 1 & 2 예측 챌린지에서 사용된 네 가지 평가 지표를 활용하였다:[18]

- Minimum Average Displacement Error(minADE) : 최적 예측 궤적과 실제 값(ground truth) 사이의 평균 L2 거리. 여기서 '최적'이란 종점 오차가 최소인 궤적을 의미한다.
- Minimum final displacement error(minFDE) : 최적 예측 궤적의 종점과 실제 값(Ground truth) 사이의 L2 거리. 여기서 '최적'이란 종점 오차가 최소인 궤적을 의미한다.
- Miss Rate(MR) : 예측된 궤적 중 어느 것도 종점 오차 기준으로 실제 값(Ground truth)에서 2.0미터 이내에 있지 않은 시나리오의 수.
- Brier Minimum Final Displacement Error(b-minFDE) :

minFDE와 유사하다. 유일한 차이점은 최적 예측 궤적의 확률을 p라고 할 때, 종점 L2 거리에$(1.0 - p)^2$를 곱한다는 것이다.

4.3 파라메터 최적화

식 (1), (2), (6) 그리고 (7)의 파라메터를 결정하기 위해 다음과 같은 접근 법을 사용하였다. Brute force 알고리즘을 적용하여 모든 파라메터를 결정하기에는 많은 시간이 소요되므로, 먼저 식 (1)과 (6)의 Threshold를 각각 전체 엣지 중 오직 70 %만이 1이 되고 나머지는 모두 0이 되도록 하였다. 다음으로 μ^a와 μ^m을 결정하기 위해 다음과 같은 접근법을 활용하였다. μ_1^m을 제외한 모든 파라메터를 0으로 고정하고 μ_1^m값을 변화시키며 최고 성능을 도출하는 값을 찾는다. 그 값을 μ_1^{m*}라 하자. 주어진 μ_1^{m*}값에 대해, 나머지 파라메터를 모두 0으로 고정하고 μ_2^m의 값을 변화시키며 최고 성능을 도출하는 파라메터를 찾는다. 이와 같은 과정을 모든 최적 파라메터 값을 찾을 때까지 반복한다. 아래의 Table 1은 QCNet을 이용한 실험 결과이다.

Table 1 Performance variation over parameters μ_1^m, μ_2^m, μ_1^a and μ_2^a. When performing the first six experiments, IGP-A2A is not applied. Given $\mu_1^m = 10$ and $\mu_2^m = 1$, the remaining six experiments are conducted. Note that QCNet is used for this experiment

μ_1^m	μ_2^m	μ_1^a	μ_2^a	b-minFDE6	minADE6	minFDE6
0	0	-	-	1.89	0.725	1.269
5	0	-	-	1.879	0.722	1.257
10	0	-	-	1.875	0.72	1.257
10	1	-	-	1.875	0.719	1.257
10	5	-	-	1.882	0.72	1.26
10	10	-	-	1.883	0.722	1.265
10	1	1	0	1.889	0.723	1.271
10	1	5	0	1.879	0.721	1.257
10	1	10	0	1.875	0.72	1.256
10	1	10	1	1.875	0.72	1.255
10	1	10	5	1.871	0.72	1.253
10	1	10	10	1.879	0.72	1.258

표에서 보듯 $\mu_1^m = 10, \mu_2^m = 1, \mu_1^a = 10$ and $\mu_2^a = 5$일 때 가장 좋은 성능을 보이는 것을 알 수 있다. 아래의 Table 2는 위 파라메터 값을 고정하고 식 (1)과 (6)의 Threshold 값을 변화시켰을 때의 QCNet의 성능을 보여준다. 앞서 언급한 바와 같이 Threshold는 전체 엣지 중 몇 % 만을 1로 둘지에 따라 결정된다.

Table 2 Performance variation over the threshold in Equation (1) and (6). Note that QCNet is used for this experiment

%	b-minFDE6	minADE6	minFDE6
30	1.96	0.741	1.331
50	1.922	0.731	1.3
70	1.871	0.72	1.253
90	1.876	0.724	1.256

표에서 보듯 전체 엣지 중 70 %만이 1이 될 때 가장 좋은 성능을 보이는 것을 알 수 있다. 이후 실험에서는 Table 1과 2의 결과로부터 얻은 최적의 파라메터를 기반으로 한 결과값임을 미리 말한다.

4.4 객관적 평가

아래 Table 3는 QCNet, HiVT, HPNet 세가지 모델에 대해 제안하는 방식을 적용했을 때 성능 변화를 보여준다. 또한 Table 4는 제안하는 방식 적용 전 후 Inference time (ms), Inference/train GPU consumption(GB)을 보여준다.

참고로 본 실험에서는 학습 시 NVIDIA RTX 4090 8개를 이용하였으며, HPNet의 경우 GPGraph와의 결합 시 학습에 필요한 메모리가 가용 가능한 GPU 메모리보다 상회하여 학습을 진행할 수 없었음을 밝힌다. 표에서 볼 수 있듯이 제안하는 방식을 적용했을 경우 Inference time이 최대 24 % 단축되고, 예측 및 학습 시 GPU 사용이 각각 최대 19 %, 30 % 감소함을 알 수 있다. 반면 성능은 근소하게 개선되거나 그대로 유지됨을 알 수 있다. 따라서 제안하는 방식은 기존의 미래 궤적 예측 모델이 실제 자율주행 차량에 적용되었을 때, 기존 모델이 제공하는 자율주행의 안전성을 해치지 않으며 동시에 메모리 사용의 감소와 처리 속도의 향상을 가져다준다고 볼 수 있다. 다만, 자율차를 이용한 실 주행 시 안전도 평가는 추가적인 시뮬레이션 평가가 필요함을 밝힌다.

한편 기존의 GP-Graph 방식은 각 방식의 성능을 오히려 떨어뜨리는 한편 속도 저하 및 메모리 사용량 증가를 초래한다. 먼저 성능이 저하되는 이유는 GP-graph는 유사한 이동 방향을 갖는 보행자들을 Grouping 하기 위한 방식으로 차량 사이의 불필요한 상호작용을 예측하는데 부적합하다. 다음으로, 특별한 네트워크를 도입하여 Grouping 과정을 학습하기 때문에 더 많은 메모리를 필요로 하고 처리 시간도 증가된다.

마지막으로 Table 5는 Threshold에 따라 QCNet+IGP의 Inference time(ms) 및 Inference/train GPU consumption(GB)이 어떻게 변하는 지를 보여주고 있다. Threshold에 의해 선택되는 엣지의 비중이 줄어 들수록 예측 속도가 빨라

Table 3 Objective evaluation on Argoverse 1 and 2 validation sets. 'GP' and 'IGP' respectively refer to 'GP-Graph' and 'IGP-Traj'

Model	Dataset	b-minFDE6	minADE6	minFDE6	MR6
QCNet	Argoverse2	1.91	0.73	1.277	0.164
QCNet+GP	Argoverse2	1.94	0.736	1.3	0.174
QCNet+IGP	Argoverse2	1.871	0.72	1.253	0.16
HiVT	Argoverse1	_	0.69	1.04	0.1
HiVT+GP	Argoverse1	-	0.7	1.07	0.11
HiVT+IGP	Argoverse1	_	0.7	1.02	0.1
HPNet	Argoverse1	1.506	0.638	0.871	0.069
HPNet+GP	Argoverse1	-	-	-	-
HPNet+IGP	Argoverse1	1.506	0.635	0.87	0.069

Table 4 Inference time(ms) and GPU memory consumption (GB) for a dense scene(Argoverse2: 158 agents, Argoverse1: 83 agents)

Model	Dataset	Inference time(ms)	GPU mem. (Inference)	GPU mem. (Train)
QCNet	Argoverse2	60.4	0.68	7.0
QCNet+GP	Argoverse2	91.7	1.99	13.5
QCNet+IGP	Argoverse2	45.8	0.54	4.88
HiVT	Argoverse1	19.3	0.09	0.46
HiVT+GP	Argoverse1	21.1	0.14	0.80
HiVT+IGP	Argoverse1	14.8	0.078	0.40
HPNet	Argoverse1	147.0	2.46	10.8
HPNet+GP	Argoverse1	-	-	-
HPNet+IGP	Argoverse1	134.0	2.03	9.36

Table 5 Inference time(ms) and GPU memory consumption (GB) over threshold for a dense scene(Argoverse2: 158 agents, Argoverse1: 83 agents). QCNet is used for the experiment

Threshold	Inference time (ms)	GPU mem (Inf, GB)	GPU mem (Train, GB)
30 %	40.9	0.453	3.91
50 %	43.0	0.480	4.38
70 %	45.8	0.547	4.88
90 %	49.2	0.615	5.39
100 %	60.4	0.681	7.0

지고 학습 및 예측에 소요되는 메모리도 줄어드는 것을 알 수 있다. 그러나 Table 3이 보여주는 바와 같이 성능도 같이 하락함을 알 수 있다. 참고로 앞서 언급한 바와 같이, IGP-Traj 구현 시 목표 에이전트 근방 15 m 이내의 에이전트 및 맵 세그먼트는 항상 엣지 값이 1이 되도록 강제하였기에 예측 속도 및 메모리 사용의 하락이 두드러지지 않음을 밝힌다.

5. 결 론

본 논문에서는 IGP-Traj라는 새로운 알고리즘을 제안하였다. 이 알고리즘은 에이전트와 도로 요소 간의 불필요한 상호작용을 제거하도록 신중하게 설계되어, Transformer 기반 궤적 예측 모델의 연산 부담을 줄일 뿐만 아니라 예측 성능을 향상시키는 것을 목표로 한다. IGP-Traj는 차량이 주로 자신 앞에 있고, 가까이 있으며, 충돌 가능성이 있는 주변 교통 에이전트 및 도로 요소에 더 많은 주의를 기울인다는 우리의 관찰에 기반한다. 우리는 제안한 방법을 기존의 세 가지 최신(SOTA) 모델에 적용하고, 대규모 실제 데이터 셋인 Argoverse1과 2에서 그 효과를 평가하였다. 실험 결과, 제안된 알고리즘은 모델의 연산 부담을 효과적으로 줄일 뿐만 아니라 예측 성능 또한 향상시킴을 보여준다.

후 기

이 논문은 2025년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임(No.RS-2024-00341055, 혼잡도로 주행 위험상황에 최적 주행행동 결정을 위한 강화학습형 자율주행 AI SW 기술 개발).

References

1) J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, C. Li and C. Schmid, "VectorNet: Encoding HD Maps and Agent Dynamics from Vectorized Representation," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

2) M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan and J. Hays, "Argoverse: 3D Tracking and Forecasting with Rich Maps," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

3) D. Choi, S.-J. Han, K. Min and J. Choi, "PathGAN: Local Path Planning with Attentive Generative Adversarial Networks," ETRI Journal, Vol.44, No.6, pp.1004-1019, 2022.

4) A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser and I. Polosukhin, "Attention Is All You Need," Advances in Neural Information Processing Systems (NIPS), 2017.

5) H.-G. Jeon, I. Bae and J.-H. Park, “Learning Pedestrian Group Representations for Multi-Modal Trajectory Prediction,” Proceedings of the European Conference on Computer Vision (ECCV), 2022.
6) J. Li, F. Yang, H. Ma, S. Malla, M. Tomizuka and C. Choi, “RAIN: Reinforced Hybrid Attention Inference Network for Motion Forecasting,” Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021.
7) L. Rowe, M. Ethier, E. Dykhne and K. Czarnecki, “FJMP: Factorized Joint Multi-Agent Motion Prediction over Learned Directed Acyclic Interaction Graphs,” Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023.
8) S. Narayanan, R. Moslemi, F. Pittaluga, B. Liu and M. Chandraker, “Divide-and-Conquer for Lane-Aware Diverse Trajectory Prediction,” Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022.
9) D. Choi and K. Min, “Vehicle Trajectory Forecasting Network Based on Static Scene Context Modulation for Autonomous Driving,” Transactions of KSAE, Vol.31, No.8, pp.597-606, 2023.
10) Y. Liu, X. Qi, E. Akin Sisbot and K. Oguchi, “Multi-Agent Trajectory Prediction with Graph Attention Isomorphism Neural Network,” Proceedings of the IEEE Intelligent Vehicles Symposium (IV), 2022.
11) T. Salzmann, B. Ivanovic, P. Chakravarty and M. Pavone, “Trajectron++: Dynamically Feasible Trajectory Forecasting with Heterogeneous Data,” Proceedings of the European Conference on Computer Vision (ECCV), 2020.
12) Z. Zhou, J. Wang, Y.-H. Li and Y.-K. Huang, “Query-Centric Trajectory Prediction,” Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp.17863-17873, 2023.
13) Z. Zhou, L. Ye, J. Wang, K. Wu and K. Lu, “HiVT: Hierarchical Vector Transformer for Multi-Agent Motion Prediction,” Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022.
14) X. Tang, M. Kan, S. Shan, Z. Ji, J. Bai and X. Chen, “HPNet: Dynamic Trajectory Forecasting with Historical Prediction Attention,” Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp.15261-15270, 2024.
15) N. Nayakanti, R. Al-Rfou, A. Zhou, K. Goel, K. S. Refaat and B. Sapp, “Wayformer: Motion Forecasting via Simple Efficient Attention Networks,” Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2023.
16) R. Child, S. Gray, A. Radford and I. Sutskever, “Generating Long Sequences with Sparse Transformers,” arXiv, 2019.
17) B. Wilson, W. Qi, T. Agarwal, J. Lambert, J. Singh, S. Khandelwal, B. Pan, R. Kumar, A. Hartnett, J. K. Pontes, D. Ramanan, P. Carr and J. Hays, “Argoverse 2: Next Generation Datasets for Self-Driving Perception and Forecasting,” Neural Information Processing Systems Datasets and Benchmarks, 2021.
18) Argoverse 2: Motion Forecasting Competition, EvalAI, https://eval.ai/web/challenges/challenge-page/1719/ evaluation, 2025.