Delayed Propagation Transformer: A Universal Computation Engine towards Practical Control in Cyber-Physical Systems

Anonymous Author(s)
Affiliation
Address
email

Abstract

Multi-agent control is a central theme in the *Cyber-Physical Systems (CPS)*. However, current control methods either receive non-Markovian states due to insufficient sensing and decentralized design, or suffer from poor convergence. This paper presents the Delayed Propagation Transformer (DePT), a new transformer-based model that specializes in the global modeling of CPS while taking into account the immutable constraints from the physical world. DePT induces a cone-shaped spatial-temporal attention prior, which injects the information propagation and aggregation principles and enables a global view. With physical constraint inductive bias baked into its design, our DePT is ready to plug and play for a broad class of multi-agent systems. The experimental results on one of the most challenging CPS – network-scale traffic signal control system in the open world – show that our model significantly outperformed the SOTA expert methods on synthetic and real-world datasets. Our codes will be fully released upon acceptance.

1 Introduction

The *Cyber-Physical System (CPS)* is ubiquitous in modern society: examples include intelligent transportation systems, power grids, autonomous automobile systems, industrial control systems, and robotics systems. A CPS consists of multiple physical agents that interact/cooperate from time to time, as well as a cyberspace that is responsible for monitoring the status of the physical system, predict future states, and/or assigns control actions to the physical agents [1, 2]. How to effectively execute multi-agent control over CPS is a problem of vital importance and is also far from being addressed.

In general, there are two kinds of methods to solve the CPS multi-agent control problem: decentralized and centralized ones. The decentralized methods usually design an individual controller for each agent based on local states. Due to the agents' continuous interaction, the features and the influence of action propagate between agents, hence the state transition for each agent relies on not only the locally observed states and local actions but also states and actions of other agents. Therefore, the locally observed states alone cannot make up a Markov Decision Process (MDP) [3, 4], which requires the controller to be able to balance the global as well as the local features [5, 6]. Existing methods tackle this issue by either complementing local information with information from the node's immediate neighbors or self history, or using Graph Convolutional Networks (GCNs) instead to recursively model information propagation and aggregation. However, none of the above methods is free of the limited localized horizon issue: even an n-layer GCNs can view at almost n-hop neighbors away. the bottleneck phenomenon that information from exponentially many neighbors is "over-squashed" into fixed-size vectors [7], etc. Hence, such methods are inherently restricted in making a complete MDP state.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

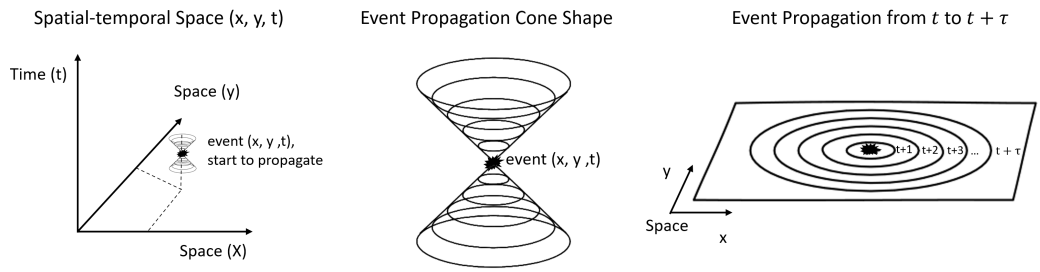


Figure 1: Illustration of the typical impact propagation pattern of an event in a *Cyber-Physical System (CPS)*. Analogous to the "light cone" (borrowed from Special Relativity), the correlation pattern in space-time follows a neighboring distribution of the cone-shaped distribution, which we set as a prior for DePT.

In contrast to decentralized methods, centralized methods consider the CPS states as a whole, hence they partially avoided the non-Markovian barrier introduced by mutual interactions. However, due to the incomplete coverage/deployment of sensors and the limitations of current sensing technology, the ideal MDP assumption still can not always be supported by only current observations in the real-world CPS [8, 9]. Given such limitations, even the snapshot of the observed node features as a whole cannot make a complete MDP state. What's more, it is hard for the centralized methods to converge due to the huge state and action spaces [10].

An additional unique barrier for CPS lies in the *direction-feature coupling* issue that makes traditional graph aggregation not suitable for CPS systems. Different from commonly adopted graph datasets such as Cora/Citeceer/Pubmed, the edges in the graph of the CPS has the physical direction, and such direction is essentially coupled with node/edge features, i.e., for edges coming from different physical directions, its features could possibly translate into permuted or even completely different meanings. Such coupling effect could cause catastrophic "feature mismatching" when one naively aggregates neighbor node features in the traditional GCN manner [11, 12].

Fortunately, the booming of transformer models shed a new light on addressing the aforementioned difficulties [13, 14, 15, 16, 17, 18, 19, 20, 21]. Transformer architectures are witnessed to be a mighty computation engine due to their global view by design. It is free of locality inductive bias and possesses a strong representation ability, making it a good candidate in the multi-agent CPS control problem. However, the traditional transformer is **not immediately ready to be plugged in** tackling CPS control problems due to the following two arising challenges:

#1. The constrained attention space due to physics.. The classical transformers benefit from their fully flexible self-attention mechanisms to capture the complex interactions within data. This free-form is desirable for vision and NLP tasks [22, 23], but no longer valuable when it comes to real CPS due to its unawareness of many physical constraints. One example is the directional propagation and direction-feature coupling issue previously mentioned. For another prominent example, the real passage of information flow between nodes and the propagation of the effect of past actions are subject to the physical propagation speed. Previous studies about graph learning using transformers mainly focus on non-physical systems and encourage between-node communication without a notion of latency [24, 25]. In CPS, however, features take time to propagate through the physical world. Without that important physical delay constraint in mind, the learned self-attention might be misled to attending to non-relevant temporal events or physically disjoint node pairs.

#2. The difficulty of training and convergence under noisy CPS data. Transformers are strong universal representers free of inductive bias, such as those in convolutions or recurrence. But the blessing of unprecedented flexibility and larger capacity can turn into the curse at training: transformers take a much longer time to converge. They are resource-heavier and much more data-hungry compared to classical convolutional, or recurrent networks [19, 21]. In CPS control problems, the training data will suffer from even higher variance due to the unavoidable randomness in state transitions. Such issue is further amplified in the multi-agent setting. How to properly train transformers to stable convergence in this scenario can present a daunting challenge.

In view of the above, we propose to model the delayed propagation effect in CPS by reinforcing the physical constraints into the transformer, via a *cone-shaped temporal-spatial prior* – that the features propagation in physical world process naturally forms the cone shape in the time-space [26, 27]. We customized a multi-level heterogeneous attention mechanism that have the causality cone prior

baked into its design. With such design, the attention between input tokens now have a physical interpretation, hence guiding the controller to learned more effectively in exploring the collaborative strategy across agents. Our main **technical innovations** are summarized below:

- We are the first to model the CPS control problem under a transformer-based framework. This is a naturally well-motivated step: transformers get ride of the agent's inherent locality restrictions, and learn global views of correlations across the whole system.
- We propose the ***Delayed Propagation Transformer (DePT)***, a new type of transformer with spatio-temporal priors baked into the attention design, to better characterize the inductive bias of physical information propagation and with better interpretability.
- We build a CPS controller based on DePT, and take the transportation signal control system as a study case. The proposed controller achieved the state-of-the-art performance in the traffic signal control task, in a large-scale network with heterogeneous node and edge types.

2 Related Works

2.1 About CPS Control Algorithms

With the booming of Reinforcement Learning (RL), learning-based control systems are generally adapted into CPS systems, including traffic network [28, 29], smart gird system [30, 31], and autonomous vehicle [32]. Using traffic network control as an example, the IntelliLight algorithm was proposed in [33]. The state and reward are a combination of several components. A multi-intersection problem was considered in [34], where an RL agent was trained for every individual intersection. The main idea is to use the pressure as the reward function, thus the algorithm is called PressLight. [35] introduced a purely localized RL agent to control traffic signals at every intersection. To efficiently model the influence of the neighbor intersections, a graph attentional network is utilized in CoLight [36]. The research in [37] showed a detailed review of conventional and RL-based methods for TSCP. To boost up the optimization process and enhance the model transferability, [38] proposed to speed up the learning process by leveraging the knowledge learned from existing scenarios. To investigate the cooperation mechanism among various types of nodes and enhance the model utility, AttendLight [39] was proposed in NeurIPS 2020. They designed a multi-level attention mechanism to handle various numbers of roads-lanes; thus enabling decision-making with different phases in an intersection. However, the previous control scheme in CPS, especially for traffic network control, either tried to model the system in ideal sensing condition, or less focused even ignored the agents' cooperation and propagation, which leads to a significant hurdle of simulation and real deployment. Although some encoding was proposed to solve the primary challenge, i.e., various lanes and phases, the cooperation between agents, global features and their propagation process have not been integrated into the decision-making procedure. To empower the CPS with the best-customized control architecture, model the agents cooperation and features propagation under noisy CPS environment in the multi-agent system is quite a necessity.

2.2 About Transformers

Transformers have witnessed great success in the application of sequential data for the discovery of input connections. In general, researchers use transformers to process the node embedding in two orthogonal directions: first, through the node-wise residual feature transformation, an arbitrary type of intra-node transformation is enabled [13, 40, 41]. Second, through the attention mechanism, features from different nodes are dynamically combined, and the inter-nodes interplays are captured [41, 42]. Previous tests have shown huge potential for transformers used in multi-agent system [43] by modeling representation of a multi-agent system by flattening connections features across time and agents. [44] used transformers to address sparse communication in multi-agent settings, which solved the data sparseness by integrating the hidden connection of inputs and achieved SOTA performance. [25] introduced transformer for the graph to seq learning in translation, which inspired us to use explicit relation encoding to allow direct cooperation between two distant nodes in agent features sharing. However, the work does not restrict feature aggregation of faraway nodes, and cannot be used in the "delayed propagation system". [45] proposed a graph transformer used for node classification/regression in heterogeneous graphs. [24] proposed a transformer-based model to learn node representations on a heterogeneous architecture. However, to the research team's best knowledge, current SOTA transformers do not have a solution for capturing both the spatial-temporal

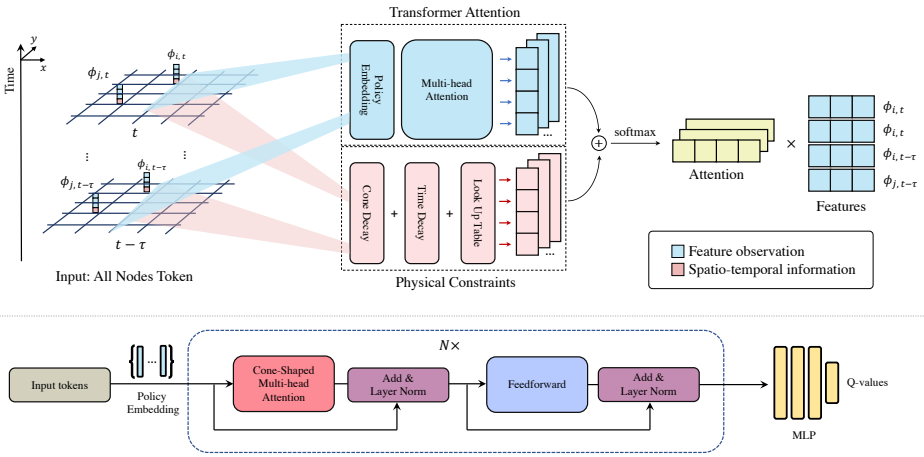


Figure 2: The DePT illustration of self-Attention layers (above) and the internal structure (below).

features and the connections encoding simultaneously, not to mention learning features under a physical shape constraint.

3 Problem Settings

The general Cyber-Physical System (CPS) can be modeled by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with a **node set** $\mathcal{V}$ and **edge set** $\mathcal{E}$. One key property that distinguishes the graph in CPS from many other graph problems is: in CPS, each individual **node** $i \in \mathcal{V}$ possesses a physical-world location $\boldsymbol{u}_{\mathrm{i}} \in \mathbb{R}^2/\mathbb{R}^3$. Such physical location exists as an immutable attribute, and exerts a constraint that is special to CPS. Unlike pure cyber world graphs such as user-product graphs where nodes can have arbitrary/instantaneous connections, the interactions between nodes in CPS are subject to their physical distances. They can only propagate under limited speeds in the physical world.

We are interested in the control problem in CPS, where each node is an agent that constantly interact with the environment as well as with each other. At every timestamp t, the agents observe the state $\boldsymbol{s}^{(t)}$ from state space $\mathcal{O}$, and take actions $\boldsymbol{a}^{(t)}$ from the action space $\mathcal{A}$. The system then get a reward $R^{(t+1)}$. The target of optimizing the CPS control is to come up with a collective optimal policy $\pi^*_{\mathcal{V}}$, under which the accumulated future reward is maximized:

$$\max_{\pi_{\mathcal{V}}} \sum_{k=0}^{\infty} \gamma^k R^{(t+k+1)} \tag{1}$$

where $\gamma \in [0, 1]$ is the discount factor. In the discrete action space settings, the agents can be trained with Q-learning types of algorithms:

$$\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}\left[R^{(t+1)} + \gamma \max_{\boldsymbol{a}^{(t+1)} \in \mathcal{A}} Q(\boldsymbol{s}^{(t+1)}, \boldsymbol{a}^{(t+1)}; \boldsymbol{\theta}) - Q(\boldsymbol{s}^{(t)}, \boldsymbol{a}^{(t)}; \boldsymbol{\theta})\right]^2, \tag{2}$$

where $\boldsymbol{\theta}$ is the parameters of the the learned Q-function $Q(\boldsymbol{s}, \boldsymbol{a}; \boldsymbol{\theta}) : \mathcal{O} \times \mathcal{A} \to \mathbb{R}$

4 DePT: Delayed Propagation Transformer

4.1 Adopt Transformer to Cyber-Physical System

We begin by introducing the transformer as the centralized agent to handle control problems in CPS. The transformer takes series of inputs, collected from all nodes spatially, and across the most recent T_{max} timestamps temporally, making up the collected inputs as: $\mathbf{X} = \{\boldsymbol{\phi}_{\mathrm{i,t}-\tau}; i \in \mathcal{V}, \tau \in \{0, 1, \cdots, T_{max}\}\}$. Every $\boldsymbol{\phi}_{\mathrm{i,t}-\tau}$ in $\mathbf{X}$ is referred to as a **token**, which can be uniquely indexed via the ID of its corresponding physical **node** and time difference relative to the current timestamp t.

For each token $\boldsymbol{\phi}_{\mathrm{i,t}-\tau}$, in addition to feed the transformer with the observed features (denoted as $\boldsymbol{f}_{\mathrm{i,t}-\tau}$), we also make our transformer aware of the policy information via the *policy embedding*: we initialize a trainable embedding matrix $\boldsymbol{P} \in \mathbb{R}^{E \times |\mathcal{A}|}$, and for each action $\boldsymbol{a}_{\mathrm{i,t}-\tau} \in \mathbb{R}^{|\mathcal{A}|}$ taken

by node i at time $t-\tau$, we index out the corresponding embedding $\boldsymbol{p}_{\mathrm{i,t}-\tau} = \boldsymbol{P}[:, \boldsymbol{a}_{\mathrm{i,t}-\tau}]$, and concatenate $\boldsymbol{p}_{\mathrm{i,t}-\tau}$ with the features $\boldsymbol{f}_{\mathrm{i,t}-\tau}$, and use $[\boldsymbol{f}_{\mathrm{i,t}-\tau} || \boldsymbol{p}_{\mathrm{i,t}-\tau}]$ as the input.

For each pair of tokens $\boldsymbol{\phi}_{\mathrm{i,t}-\tau}$ and $\boldsymbol{\phi}_{\mathrm{j,t}-\rho}$, where $i, j \in \mathcal{V}$ and $0 \leq \tau, \rho \leq T_{max}$, the traditional design of transformer will compute the pre-softmax attention via:

$$a^{(T)}\left(\boldsymbol{\phi}_{\mathrm{i,t}-\tau}, \boldsymbol{\phi}_{\mathrm{j,t}-\rho}\right) = \boldsymbol{\phi}_{\mathrm{i,t}-\tau}^{\top} \boldsymbol{W}_Q^{\top} \boldsymbol{W}_K \boldsymbol{\phi}_{\mathrm{j,t}-\rho} \tag{3}$$

where $\boldsymbol{W}_K$ and $\boldsymbol{W}_Q$ are weight matrices to compute the key and query component, respectively. For every token $\boldsymbol{\phi}_{\mathrm{i,t}-\tau}^{(l)}$ at the l-th layer, its output from an attention layer is

$$\boldsymbol{\phi}'_{\mathrm{i,t}-\tau} = \sum_{j \in \mathcal{V}, \rho \in [T_{max}]} e\left(\boldsymbol{\phi}_{\mathrm{i,t}-\tau}, \boldsymbol{\phi}_{\mathrm{j,t}-\rho}\right) \boldsymbol{W}_V \boldsymbol{\phi}_{\mathrm{j,t}-\rho} \tag{4}$$

where $e\left(\boldsymbol{\phi}_{\mathrm{i,t}-\tau}, \boldsymbol{\phi}_{\mathrm{j,t}-\rho}\right) = \mathrm{softmax}_{j \in \mathcal{V}, \rho \in [T_{max}]}\left(a^{(f)}\left(\boldsymbol{\phi}_{\mathrm{i,t}-\tau}, \boldsymbol{\phi}_{\mathrm{j,t}-\rho}\right)/\Gamma\right)$, and Γ is the temperature factor, the weight matrix $\boldsymbol{W}_V$ maps the input to the value component, and $e(\cdot,\cdot)$ denotes the pairwise post-softmax attention. In practice, we adopted multi-head attention blocks. Each attention head will have an output $\boldsymbol{\phi}'^{(k)}, k = 1, \cdots, N$, where N is the number of attention heads. The computed attention heads are concatenated together and remapped to the output dimension via a linear block. Following the attention layer, we append a layer normalization module with a skip connection. Subsequently, the output will be fed into a Feed-Forward Network (FFN) followed by a skip-connected layer normalization block. Multi-head attention, normalization, and FFN combine to be a single encoding layer. We stack such layers building a L-layer transformer. At the output layer, we only read out the feature embedding of the nodes at the current time t, i.e., $\boldsymbol{\phi}_{\mathrm{i,t}-\tau}, \forall i \in \mathcal{V}$. Afterward, we employ another fully connected layer to map the feature embedding to the pre-action q-values. The architecture are illustrated in Fig. 2.

4.2 Reinforce Spatial-temporal Constrains

As illustrated in Fig.1, the spatial-temporal token pairs that have strong connections are more likely to be located around the cone-shaped area. We argue that to capture the most relevant spatial-temporal attention, the transformer agent operating in such settings should have their attention obey some neighboring distribution of the cone-shaped prior.

To reinforce such prior attention distribution, we train a new function $a^{(D)}(\boldsymbol{\phi}, \boldsymbol{\phi}')$ (D for DePT), in addition to the Traditional transformer attention $a^{(T)}$ in Eq.3. These two functions are added up to obtain the pre-softmax attention value. $a^{(D)}(\boldsymbol{\phi}, \boldsymbol{\phi}')$ consists three components: the learned pair-relation Look-Up-Table embedding $\lambda^{(attn)}[i, j]$ (**LUT**), the cone shaped correlation decay function $\gamma(\epsilon_{\mathrm{j},\rho;\mathrm{i},\tau})$ (**ConeDecay**), the temporal correlation decay function $\sigma(\tau - \rho)$ (**TimeDecay**). Below we specify them in detail.

LUT. To capture the location-associated patterns, we enable our DePT to learn a scalar attention value $\lambda^{(attn)}[i, j]$ for every pair of nodes (i, j) (tokens associated to the same nodes but with different timestamp share the same embedding). Such LUT is learned independently for every attention head in the original transformer.

ConeDecay. In the physical part of CPS, both the effect of nodes' actions and the passage of features can be modeled as *information flow* that are constantly exchanging/propagating between nodes. Think of a information flow with speed v that originates from node j at time $t-\rho$, and heads for node i at time $t-\tau$. This flow will induce a causal connection in the physical world between two associated tokens $\boldsymbol{\phi}_{\mathrm{i,t}-\tau}$ and $\boldsymbol{\phi}_{\mathrm{j,t}-\rho}$. It is straightforward to define the causal deviation as follows:

$$\epsilon_{\mathrm{j},\rho;\mathrm{i},\tau}(\boldsymbol{\phi}_{\mathrm{j,t}-\rho} \rightarrow \boldsymbol{\phi}_{\mathrm{i,t}-\tau}) = (\tau - \rho)v - \|\boldsymbol{u}_{\mathrm{i}} - \boldsymbol{u}_{\mathrm{j}}\| \tag{5}$$

Under this definition, $\epsilon_{\mathrm{j},\rho;\mathrm{i},\tau} < 0$ and $\epsilon_{\mathrm{j},\rho;\mathrm{i},\tau} > 0$ corresponds to the expected "past" and "future" for the incident "$\boldsymbol{\phi}_{\mathrm{j,t}-\rho} \rightarrow \boldsymbol{\phi}_{\mathrm{i,t}-\tau}$ is exactly reachable by flow with speed v". We use the function $\gamma(\epsilon_{\mathrm{j},\rho;\mathrm{i},\tau}; W_\gamma)$ parameterized by W_γ to penalize those "unreachable" token pairs, hence $\gamma(\cdot)$ is expected to obtain its global maximum at $\epsilon_{\mathrm{j},\rho;\mathrm{i},\tau} = 0$, and is a bi-directional decreasing function. It's gradient $\left|\frac{\mathrm{d}\gamma(\epsilon_{\mathrm{j},\rho;\mathrm{i},\tau_{ij}})}{\mathrm{d}\epsilon_{\mathrm{j},\rho;\mathrm{i},\tau_{ij}}}\right|$ depicts how fast the attention should drop and how "wide" the attention should

spread, for the token $\boldsymbol{\phi}_{\mathrm{j,t}-\rho}$ that deviates from the cone of $\boldsymbol{\phi}_{\mathrm{i,t}-\tau}$. For the estimation of the speed, v is the average of three learnable functions: the origination/destination speed, and a speed LUT (different from the above $\lambda^{(attn)}$): $\hat{v} = \frac{1}{3}\left(\nu^{(o)}(\boldsymbol{\phi}'_{\mathrm{j,t}-\rho}; W_{\nu^{(o)}}) + \nu^{(d)}(\boldsymbol{\phi}'_{\mathrm{i,t}-\tau}; W_{\nu^{(d)}}) + \lambda^{(v)}[i,j]\right)$

TimeDecay. Information from more distant past should be payed with less attention. To fit such time dissipation effect, given two tokens $\boldsymbol{\phi}_{\mathrm{i,t}-\tau}, \boldsymbol{\phi}_{\mathrm{j,t}-\rho}$, we let our DePT learn an additive function $\sigma(\tau-\rho; W_\sigma)$ parameterized by W_σ for the pairwise attention.

Overall Attention. The ConeDecay, TimeDecay, and LUT are all learned separately for every transformer encoder block and every attention head. To further guide the capturing of desired attention as well as accelerate the training/inference in two-fold speed, we mask out the attention between token pairs ($\boldsymbol{\phi}_{\mathrm{i,t}-\tau}$, $\boldsymbol{\phi}_{\mathrm{j,t}-\rho}$) whenever $\rho > \tau$, that is, not letting node $\boldsymbol{n}_{\mathrm{i}}$ pay attention to node $\boldsymbol{n}_{\mathrm{j}}$ when $\boldsymbol{n}_{\mathrm{j}}$ is in the future timestamp of $\boldsymbol{n}_{\mathrm{i}}$ and have no chance to pass information to $\boldsymbol{n}_{\mathrm{i}}$. Distilling the essence up to this point, the architectures of the proposed DePT encoder blocks is shown in Fig. 2, and the pre-softmax attention between token pair ($\boldsymbol{\phi}_{\mathrm{i,t}-\tau}$, $\boldsymbol{\phi}_{\mathrm{j,t}-\rho}$) is computed via:

$$Attn(\boldsymbol{\phi}_{\mathrm{i,t}-\tau}, \boldsymbol{\phi}_{\mathrm{j,t}-\rho}) = \begin{cases} -\infty & \text{, if } \rho > \tau \\ \gamma(\epsilon_{\mathrm{j},\rho;\mathrm{i},\tau}) + \sigma(\tau-\rho) + \lambda^{(attn)}[i,j] & \\ \quad + \boldsymbol{\phi}^{\mathrm{T}}_{\mathrm{i,t}-\tau} W_Q^{\mathrm{T}} W_K \boldsymbol{\phi}_{\mathrm{j,t}-\rho} & \text{, Otherwise} \end{cases} \tag{6}$$

4.3 Speeding up the training of DePT with Prior Pre-fitting and Imitation Learning

Same as all other centralized control methods, transformers are not immune to slow convergence issues during RL. Luckily, DePT is born with good attention prior to capture the underlying feature correlation easily, and to pay desired attention, take the desired action even at the early training stage. To boost the potential of such prior to the extreme, we propose to pre-train the ConeDecay, TimeDecay, LUT components of DePT before interacting with the environments. In addition to pre-train the priors of DePT, we also adopt the imitation learning trick in the early stage of reinforcement learning.

Pre-fitting the Priors. There are 6 learnable components in every group of priors: the ConeDecay function $\gamma(\cdot)$, three speed estimators to compute $\epsilon_{\mathrm{j},\rho;\mathrm{i},\tau}$ for the ConeDecay: $\nu^{(o)}(\cdot), \nu^{(d)}(\cdot), \lambda^{(v)}[\cdot,\cdot]$, the TimeDecay function $\sigma(\cdot)$, and the attention LUT $\lambda^{(attn)}[\cdot,\cdot]$. Above them, the attention LUT and the speed LUT are matrices and can be initialized as M_0 and $M_{\bar{v}}$ respectively, where $M_x(x=0,\bar{v})$ stands for a random matrix with all entries i.i.d. and have mean value x, and $\bar{v}$ is the average flow speed according to the statistics of the system. $\gamma(\cdot)$ and $\sigma(\cdot)$ are univariate functions, and can be pre-trained to fit certain desired analytical functions. In practice, we use $y = -kx^2$ to pre-fit $\gamma(\cdot)$ and $\sigma(\cdot)$, where k is the normalization factor depending on the variance of input features. The two last functions $\nu^{(o)}(\cdot), \nu^{(d)}(\cdot)$, though multi-variate, can still be easily pre-fitted to labels sampled from $\mathcal{N}(\bar{v}, 0.1)$, and also lead to good initializations. All the above pre-fittings/initializations involve no huge tensor computation, and can therefore be accomplished within minute-level CPU computation time. However, such pre-fittings/initializations can largely boost the convergence of DePT in the reinforcement learning stages, as to be witnessed in section 6.1.

Warm-up with Imitation Learning. Since the decentralized controller is easier to converge than the centralized method, we adopt the behavior cloning/imitation learning(IL) [46] as the warm-up stage of RL. We divide the entire RL into the IL stage and the independent training stage. Before RL, a decentralized baseline controller is first trained and acts as the teacher model. During the IL stage, the actions are taken by DePT and evaluated by the teacher model, for the purpose of behavior cloning. Then during the following independent training stage, the DePT takes actions and evaluates the Q-values by itself.

5 Experiments and Discussions

We use the network-wide urban traffic signal control (TSC) problem under connected and autonomous vehicles (CAVs) environment as a CPS control example to illustrate the performance of DePT. CAVs have great promises to help improve the performance of traffic signal control system, since they can provide traffic information (i.e., vehicle position, speed, acceleration, etc.) that are crucial to

determining the signal phases and timings. The network-wide TSC under CAV is a multi-agent control problem where distributed control methods will encounter the non-MDP property for a single agent. In contrast, centralized control methods usually suffer from inefficient learning and slow convergence. In this section, we verify the ability of DePT under this challenging setting. We first define the TSC problem in Section 5.1 and introduce the dataset and simulation tools in Section 5.2. Then, we show the simulation results and discuss them in Section 5.3.

5.1 Network-wide Urban TSC problem

In the TSC setting, the graph is materialized into the urban traffic network, where the nodes/agents represent intersections and edges indicate the connecting roads between intersections. The viable action at each node is the traffic light signal to control the traffic flow. For a specific node, we take the surrounded traffic information provided by CAVs as the state, and use the signal phases as actions. Specifically, assume there is k_i incoming lanes for intersection i, we define the state of this intersection as:

- $curTime \in \mathbb{R}$: current time in simulation;
- $numIn \in \mathbb{N}^{k_i}$: the number of CAVs on each of the k_i lanes;
- $numQue \in \mathbb{N}^{k_i}$: the number of stopped CAVs (i.e., those CAVs with speeds $< 0.1m/s$) on each of the k_i lanes;

For intersection i, we define the action space as $a_i = \{1, 2, ...n_i\}$ where n_i is the total number of phases for intersection i. As an example, consider an intersection i that has two phases (i.e., $n_i = 2$): {West→East, East→West} and {South→North, North→South}, we define $a_i = 1$ as West→East/East→West green and the other conflict phase red, $a_i = 2$ as South→North/North→South green and the other conflict phase red.

5.2 Dataset

We use Cityflow [47] as the simulation platform due to the high computation efficiency. To evaluate the performance of DePT, we run experiments on both synthetic and real urban traffic networks built by [36]. The data is for scientific research usages, which is open-source and does not contain offensive content.

For the synthetic traffic networks, we use a 6×6 synthetic grid network with 36 agents taking actions every 10 seconds. There are 12 movements associated with every intersection (i.e., West→East, West→North, West→South, East→West, East→South, East→North, South→North, South→West, South→East, North→South, North→East, North→West). We design four phases for each intersection: {West→East, West→South, East→West, East→North}, {West→North, East→South}, {South→North, South→East, North→South, North→West} and {South→West, North→East}. There are 12 incoming lanes for each intersection, and the length of each lane is $300m$. Similar to [36], we test two flow settings for this network: bi-directional (noted as Grid-Bi) and uni-directional (noted as Grid-Uni). In Grid-Bi, we generate 300 vehicles/lane/hour for West→East and East→West movements and 90 vehicles/lane/hour for South→North and North→South movements. In Grid-Uni, we generate the same flows, but only for West→East and North→South movements.

For the real-world scenarios, we use the upper east side of Manhattan, New York, with 28×7 active agents as shown in Figure 3. The road network is imported from OpenStreetMap, and the flow is extracted by taxi data. For the evaluation metrics, we use the average travel time (noted as AvgTT, in seconds) of all vehicles and the average queue length (noted as AvgQue, in vehicles) during the experiments.

5.3 Experimental Settings and Results

The number of different timestamps to be feed into the DePT (T_{max}) is set to be 10 in our experiments, with a sample rate of 0.1 Hz (every two consequent timestamps have 10 seconds interval). The DePT is trained for 200 rounds for each dataset, while the first 100 rounds are imitation learning and the second 100 rounds are independent training. During each round, we run simulations for 4 simulation hours and record state/action pairs, then we train DePT for 100 epochs with Double-DQN using the newly generated data. The DePT is compared with three typical network-wide urban TSC methods:

- Fixed-time [48]: A classical signal planning method that uses daily vehicle volumes to calculate pre-timed signal plans based on operation manuals. The signal plans will not change once they are generated.
- Max-pressure [49]: A network-wide signal control method that dynamically generates greedy signal plans by maximizing the "pressure" (calculated from queue length) of upstream and downstream roads.
- Co-light [36]: A state-of-art multi-agent RL-based network-wide signal control method that uses graph attentional networks to incorporate the temporal and spatial impacts among different intersections.

The teacher model we use during the IL stage is the Co-light model. The results for all models are shown in Table 1. As can be observed, DePT achieves the best performance among the four models, and it especially outperforms its teacher model Co-light.

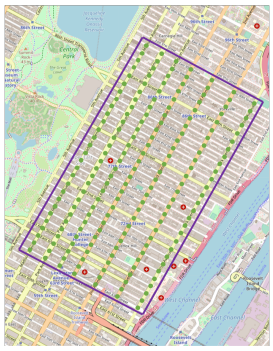


Figure 3: The Road Map of New York Dataset

Dataset	Grid-Bi		Grid-Uni		NewYork	
	AvgQue	AvgTT	AvgQue	AvgTT	AvgQue	AvgTT
Fixed-time	0.11	225.62	0.06	225.62	1.72	1957.35
Max-pressure	0.09	208.13	0.04	198.93	1.75	1628.64
Co-light	**0.05**	191.66	**0.04**	188.23	1.69	1605.56
DePT	**0.05**	**184.22**	**0.04**	**180.71**	**1.52**	**1555.47**

Table 1: Evaluation results of DePT against others.

Stages	Imitation Learning				Independent Training			
Rounds	0	30	60	90	100	130	160	190
DePT	1143.1	**244.3**	**211.6**	**202.4**	**199.6**	**188.2**	**185.0**	**184.9**
no-pre-fit	**997.2**	782.1	243.8	218.9	216.2	192.4	190.1	189.2
no-ConeDecay	1042.6	616.4	317.3	227.7	225.2	240.3	285.4	269.3
TTE	1215.4	1418.9	955.8	326.7	299.5	265.9	297.1	313.7

Table 2: Ablation study results: AvgTT on Grid-Bi. TTE refer to the traditional transformer encoder.

6 Ablation Studies and Model Interpretations

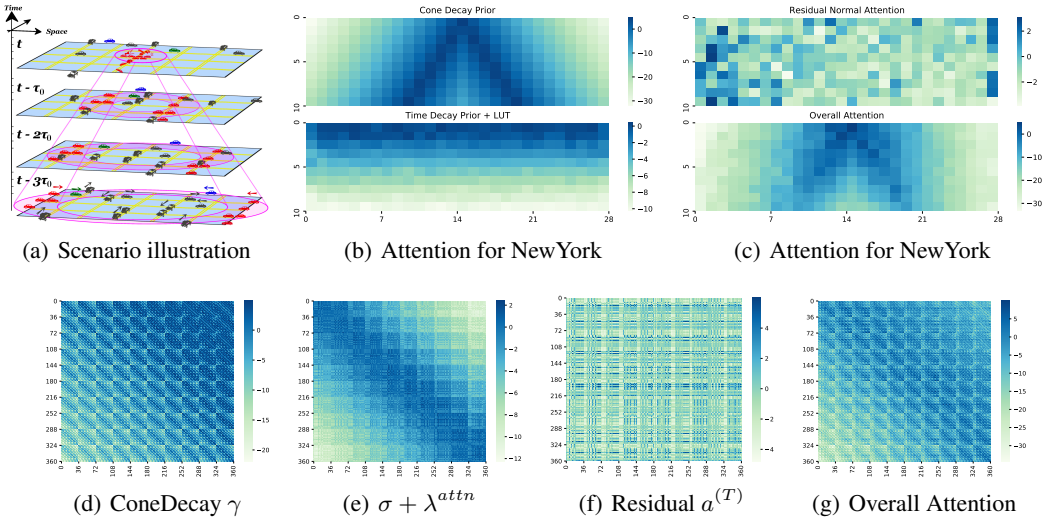
6.1 Ablation Studies

We made ablation studies of DePT on two aspects: the effect of the low-budget pre-fitting of priors as described in section 4.3, and the existence of newly proposed priors. Since the convergence rate is substantially slower if without imitation learning, all the experiments are executed with imitation learning. The AvgTT results on Grid-Bi dataset are shown in Table 2. In the table, the "DePT" corresponds to the proposed DePT that has its priors pre-fitted, where the "no-pre-fit" corresponds to the DePT without pre-fitting prior, the "no-ConeDecay" is the DePT without ConeDecay component but still pre-fitted other prior, and the TTE corresponds to the traditional transformer encoder without any prior. As can be observed, the "DePT" achieves the fastest convergence speed as well as the best performance, where the "no-pre-fit" model also slowly converged to satisfactory performance. However, when the ConeDecay component is removed, both the "no-ConeeDecay" and the TTE model fail to converge after entering the independent training stage, possibly due to loose of attention focus.

6.2 Model Visualizations

The attention patterns are visualized in figure 4. Subfigure (a) is a visualization of the urban grid roadmap, while subfigures (b)-(g) are sampled attention matrices at the 30-th timestamp, coming from the 1st attention head at the last transformer encoder block. In all subfigures in (b)-(g), we display the attention as 4 parts: ❶ the ConeDecay prior $\gamma(\cdot)$, ❷ the TimeDecay prior $\sigma(\cdot)$ plus the attention LUT $\lambda^{(attn)}[\cdot,\cdot]$, ❸ the traditional attention $a^{(T)}(\cdot)$ as a residual component, and ❹ the overall attention, which is the summation of the previous three components.

Denote the size of the grid as $h \times w$, then number of tokens is $N_{token} = T_{max} \times h \times w$. The subfigures (d)(e)(f)(g) are the $N_{token} \times N_{token}$ attention matrix $\boldsymbol{A}$ itself, coming from the Grid-Bi dataset. The attention matrix in (b)(c) come from the NewYork dataset, and are sampled in the following way: we pick the certer node n_0 of the 7×28 grid of Newyork, then select all 28 nodes



(a) Scenario illustration (b) Attention for NewYork (c) Attention for NewYork

(d) ConeDecay γ (e) $\sigma + \lambda^{attn}$ (f) Residual $a^{(T)}$ (g) Overall Attention

Figure 4: Visualized Attention Distributions

that are in the same verticle line as n_0. Then we extract the attention from $\boldsymbol{A}$ that correspond to the attention n_0 paid to all these 28 nodes in the past T_{max} timestamps, forming a sub-matrix of $\boldsymbol{A}$ of shape $T_{max} \times h$ (which is reshaped from the union of multiple discontinuous segments in certain row of $\boldsymbol{A}$).

As can be seen from the figures, the overall learned attention pattern displays the decay effect both along the cone-deviating direction and along the time axis, which is the same as it is designed for. Observing the cone attention in subfigure (b) (the above one), the learned cause decay effect shows asymmetry: the outwards direction (relative future) decayed faster than the inwards direction (relative past), which is in line with the underlying Poisson distribution for the arrival of traffic flows. Note that such attention patterns rely on the instant dynamic features, and hence can vary across time, center tokens considered, and across different encoder blocks. Such observations demonstrate the interpretability of our DePT.

7 Conclusions and Broad Impacts

In this paper, our research team proposed a new transformer-based model for feature learning and control in CPS – the Delayed Propagation Transformer (DePT). The DePT induces cone-shaped spatial-temporal attention priors into the transformer, which help boost the transformer's ability to capture the desired attention pattern as ruled by immutable physical constraints. We choose one of the most complicated open-word system – traffic network to test the potential of performance of DePT on both synthetic and real open-source datasets, and our result shows that our model not only converges faster and better than the traditional transformer but also outperforms other expert neural networks for traffic network control problem. With the DePT, the researchers can target to solve the CPS control scheme from a new perspective – model physical constraints by sequential inputs in the multi-agent CPS by a centralized framework, in an end-to-end approach.

The technical and practical broad impact can be summarized into several bullets. Firstly, as a successful global-view controller, DePT offers a centralized control scheme that has the potential to solve the control problem in a system with limited sensing. Secondly, the newly proposed DePT is a general module designed for a system with sparsity/speed limit as physical constraints, and is ready to plug-and-play for many other cyber-physical systems, such as the electrical power grids, industrial control systems, as well as the e-commerce delivery systems. Finally, the proposed imitation learning and prior pre-fitting help to release the slow convergence hurdle for centralized controllers. We do not believe that this research place any significant risk of societal harm, since it is scientific in nature. Our findings are highly likely to boost the multi-agent control research in various applications.

References

[1] Stefano Zanero. Cyber-physical systems. *Computer*, 50(4):14–16, 2017.

[2] Yang Liu, Yu Peng, Bailing Wang, Sirui Yao, and Zihe Liu. Review on cyber-physical systems. *IEEE/CAA Journal of Automatica Sinica*, 4(1):27–40, 2017.

[3] Jayesh K Gupta, Maxim Egorov, and Mykel Kochenderfer. Cooperative multi-agent control using deep reinforcement learning. In *International Conference on Autonomous Agents and Multiagent Systems*, pages 66–83. Springer, 2017.

[4] Thomas Morstyn, Branislav Hredzak, and Vassilios G Agelidis. Cooperative multi-agent control of heterogeneous storage devices distributed in a dc microgrid. *IEEE Transactions on Power Systems*, 31(4):2974–2986, 2015.

[5] Tianshu Chu, Jie Wang, Lara Codecà, and Zhaojian Li. Multi-agent deep reinforcement learning for large-scale traffic signal control. *IEEE Transactions on Intelligent Transportation Systems*, 21(3):1086–1095, 2019.

[6] Teodora Sanislav, Sherali Zeadally, George Mois, and Hacène Fouchal. Multi-agent architecture for reliable cyber-physical systems (cps). In *2017 IEEE Symposium on Computers and Communications (ISCC)*, pages 170–175. IEEE, 2017.

[7] Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. *arXiv preprint arXiv:2006.05205*, 2020.

[8] Matthijs TJ Spaan. Partially observable markov decision processes. In *Reinforcement Learning*, pages 387–414. Springer, 2012.

[9] Khac Duc Do. Bounded controllers for formation stabilization of mobile agents with limited sensing ranges. *IEEE Transactions on Automatic Control*, 52(3):569–576, 2007.

[10] Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John Agapiou, Julian Schrittwieser, et al. Starcraft ii: A new challenge for reinforcement learning. *arXiv preprint arXiv:1708.04782*, 2017.

[11] Frank L Lewis, Hongwei Zhang, Kristian Hengster-Movric, and Abhijit Das. *Cooperative control of multi-agent systems: optimal and adaptive design approaches*. Springer Science & Business Media, 2013.

[12] Hashim A Hashim, Sami El-Ferik, and Frank L Lewis. Neuro-adaptive cooperative tracking control with prescribed performance of unknown higher-order nonlinear multi-agent systems. *International Journal of Control*, 92(2):445–460, 2019.

[13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[14] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. *arXiv preprint arXiv:2012.07436*, 2020.

[15] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *International Conference on Machine Learning*, pages 4055–4064. PMLR, 2018.

[16] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. A survey on visual transformer. *arXiv preprint arXiv:2012.12556*, 2020.

[17] Igor V Tetko, Pavel Karpov, Ruud Van Deursen, and Guillaume Godin. State-of-the-art augmented nlp transformer models for direct and single-step retrosynthesis. *Nature communications*, 11(1):1–11, 2020.

[18] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

[19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[20] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020.

[21] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[22] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *arXiv preprint arXiv:2101.01169*, 2021.

[23] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. *arXiv preprint arXiv:2009.06732*, 2020.

[24] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. Graph transformer networks. *arXiv preprint arXiv:1911.06455*, 2019.

[25] Deng Cai and Wai Lam. Graph transformer for graph-to-sequence learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7464–7471, 2020.

[26] N Christ, B Hasslacher, and Alfred H Mueller. Light-cone behavior of perturbation theory. *Physical Review D*, 6(12):3543, 1972.

[27] Marc Cheneau, Peter Barmettler, Dario Poletti, Manuel Endres, Peter Schauß, Takeshi Fukuhara, Christian Gross, Immanuel Bloch, Corinna Kollath, and Stefan Kuhr. Light-cone-like spreading of correlations in a quantum many-body system. *Nature*, 481(7382):484–487, 2012.

[28] Xiaoyuan Liang, Xunsheng Du, Guiling Wang, and Zhu Han. A deep reinforcement learning network for traffic light cycle control. *IEEE Transactions on Vehicular Technology*, 68(2):1243–1253, 2019.

[29] Wade Genders and Saiedeh Razavi. Using a deep reinforcement learning agent for traffic signal control. *arXiv preprint arXiv:1611.01142*, 2016.

[30] Jakob Stoustrup, Anuradha Annaswamy, Aranya Chakrabortty, and Zhihua Qu. Smart grid control: overview and research opportunities. 2018.

[31] M Ourahou, W Ayrir, B EL Hassouni, and A Haddi. Review on smart grid control and reliability in presence of renewable energies: Challenges and prospects. *Mathematics and Computers in Simulation*, 167:19–31, 2020.

[32] Sampo Kuutti, Richard Bowden, Yaochu Jin, Phil Barber, and Saber Fallah. A survey of deep learning applications to autonomous vehicle control. *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[33] Hua Wei, Guanjie Zheng, Huaxiu Yao, and Zhenhui Li. Intellilight: A reinforcement learning approach for intelligent traffic light control. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2496–2505, 2018.

[34] Hua Wei, Chacha Chen, Guanjie Zheng, Kan Wu, Vikash Gayah, Kai Xu, and Zhenhui Li. Presslight: Learning max pressure control to coordinate traffic signals in arterial network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1290–1298, 2019.

[35] Chacha Chen, Hua Wei, Nan Xu, Guanjie Zheng, Ming Yang, Yuanhao Xiong, Kai Xu, and Zhenhui Li. Toward a thousand lights: Decentralized deep reinforcement learning for large-scale traffic signal control. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3414–3421, 2020.

[36] Hua Wei, Nan Xu, Huichu Zhang, Guanjie Zheng, Xinshi Zang, Chacha Chen, Weinan Zhang, Yanmin Zhu, Kai Xu, and Zhenhui Li. Colight: Learning network-level cooperation for traffic signal control. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1913–1922, 2019.

[37] Hua Wei, Guanjie Zheng, Vikash Gayah, and Zhenhui Li. A survey on traffic signal control methods. *arXiv preprint arXiv:1904.08117*, 2019.

[38] Xinshi Zang, Huaxiu Yao, Guanjie Zheng, Nan Xu, Kai Xu, and Zhenhui Li. Metalight: Value-based meta-reinforcement learning for traffic signal control. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1153–1160, 2020.

[39] Afshin Oroojlooy, Mohammadreza Nazari, Davood Hajinezhad, and Jorge Silva. Attendlight: Universal attention-based reinforcement learning model for traffic signal control. *arXiv preprint arXiv:2010.05772*, 2020.

[40] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal transformers. *arXiv preprint arXiv:1807.03819*, 2018.

[41] Chen-Hsuan Lin, Ersin Yumer, Oliver Wang, Eli Shechtman, and Simon Lucey. St-gan: Spatial transformer generative adversarial networks for image compositing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9455–9464, 2018.

[42] Matthew CH Lee, Ozan Oktay, Andreas Schuh, Michiel Schaap, and Ben Glocker. Image-and-spatial transformer networks for structure-guided image registration. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 337–345. Springer, 2019.

[43] Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris Kitani. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. *arXiv preprint arXiv:2103.14023*, 2021.

[44] Jeevana Priya Inala, Yichen Yang, James Paulos, Yewen Pu, Osbert Bastani, Vijay Kumar, Martin Rinard, and Armando Solar-Lezama. Neurosymbolic transformers for multi-agent communication. *arXiv preprint arXiv:2101.03238*, 2021.

[45] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. Heterogeneous graph transformer. In *Proceedings of The Web Conference 2020*, pages 2704–2710, 2020.

[46] Yunzhu Li, Jiaming Song, and Stefano Ermon. Infogail: Interpretable imitation learning from visual demonstrations. *arXiv preprint arXiv:1703.08840*, 2017.

[47] Huichu Zhang, Siyuan Feng, Chang Liu, Yaoyao Ding, Yichen Zhu, Zihan Zhou, Weinan Zhang, Yong Yu, Haiming Jin, and Zhenhui Li. Cityflow: A multi-agent reinforcement learning environment for large scale city traffic scenario. In *The World Wide Web Conference*, pages 3620–3624, 2019.

[48] Peter Koonce and Lee Rodegerdts. Traffic signal timing manual. Technical report, United States. Federal Highway Administration, 2008.

[49] Pravin Varaiya. The max-pressure controller for arbitrary networks of signalized intersections. In *Advances in Dynamic Network Modeling in Complex Transportation Systems*, pages 27–66. Springer, 2013.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] abstract summarizes accurately
   (b) Did you describe the limitations of your work? [Yes] Refer to introduction section
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] Yes, refer to the last section. To the research team's best knowledge, there are no potential negative societal can be found.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A] No theoretical results
   (b) Did you include complete proofs of all theoretical results? [N/A] No theoretical results
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The codes are included in the supplemental materials
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] Refer to the experiments section
   (b) Did you mention the license of the assets? [Yes] We cited the creator's paper, which contains the license by itself
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] All the data/simulation module we use are open source and publicly available.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] Refer to the experiments section.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]