A APPENDIX

In this Appendix, we will introduce the data sheet of XTraffic, the statement of responsibility, more details of XTraffic data, related work of the 4 traffic tasks, and the experiment settings of the Post-Incident Traffic Forecasting, Incident Classification, Global Causal Analysis, and Local Causal Analysis.

A.1 DATA SHEET OF XTRAFFIC

In this section, we follow the datasheet format (**?**) to answer the critical questions to a standard dataset.

A.1.1 MOTIVATION

- **For what purpose was the dataset created?** The XTraffic is the most recent in terms of the collection period and contains the largest number of sensors, covering three distinct types of traffic volume. This ensures the timeliness of traffic research, providing a robust foundation for studies aiming to capture and explain traffic dynamics, causation, and interrelations. XTraffic serves as a rigid testing bed and empirical support to justify model effectiveness and interoperability in deep learning and the traffic community.
- **Who created the dataset?** The Machine Intelligence and kNowledge Engineering (MINE) lab.
- **Who funded the creation of the dataset?** The creation of the dataset and research reported in this paper was supported by funding from King Abdullah University of Science and Technology (KAUST).

A.1.2 COMPOSITION

- **What do the instances that comprise the dataset represent.** See the Section 3 Data Introduction.
- **How many instances are there in total?** For traffic time series data, the total number of instances is 105120 (Time Slots Number) $\times$ 16,972 (Sensor Number) $\times$ 3 (Feature Number). For incidents, the instances is 476,766.
- **Does the dataset contain all possible instances or is it a sample (not necessarily random) of instances from a larger set?** All possible instances excluding the sensors with a large number of missing values.
- **What data does each instance consist of?** See the Section 3 Data Introduction.
- **Is there a label or target associated with each instance?** See the Section 3 Data Introduction.
- **Is any information missing from individual instances?** Raw data missing.
- **Are relationships between individual instances made explicit** Yes, they are connected by time, location, and sensor ID.
- **Are there recommended data splits?** For Traffic forecasting, we recommend the ratio of 6:2:2 for training, valid, and test dataset. It is a common setting (Shao et al., 2022; Guo et al., 2019).
- **Are there any errors, sources of noise, or redundancies in the dataset?** Yes. The traffic time series are collected from sensors, and it may not count all of the passing vehicles.
- **Is the dataset self-contained, or does it link to or otherwise rely on external resources?** No.
- **Does the dataset contain data that might be considered confidential?** No.
- **Does the dataset contain data that, if viewed directly, might be offensive, insulting, threatening, or might otherwise cause anxiety?** No.

A.1.3 Collection Process

- **How was the data associated with each instance acquired?** The data is directly observable.
- **What mechanisms or procedures were used to collect the data (e.g., hardware apparatuses or sensors, manual human curation, software programs, software APIs)?** We use the PeMS data table corresponding URLs to collect the data.
- **If the dataset is a sample from a larger set, what was the sampling strategy (e.g., deterministic, probabilistic with specific sampling probabilities)?** Not fit.
- **Who was involved in the data collection process (e.g., students, crowdworkers, contractors) and how were they compensated (e.g.,how much were crowdworkers paid)?** No person is involved in the collection process.
- **Over what timeframe was the data collected?** The data was collected from April 20, 2024, to May 10, 2024. The dataset covers the entire year of 2023.
- **Were any ethical review processes conducted (e.g., by an institutional review board)?** No.

A.1.4 Preprocessing/cleaning/labeling

- **Was any preprocessing/cleaning/labeling of the data done (e.g., discretization or bucketing, tokenization, part-of-speech tagging, SIFT feature extraction, removal of instances, processing of missing values)?**Yes. We construct the adjacency matrix for sensors in the road network.
- **Was the "raw" data saved in addition to the preprocessed/cleaned/labeled data (e.g., to support unanticipated future uses)?** Yes.
- **Is the software that was used to preprocess/clean/label the data available?** The code is released in our GitHub repository `https://xaitraffic.github.io/`

A.1.5 Uses

- **Has the dataset been used for any tasks already?** No.
- **Is there a repository that links to any or all papers or systems that use the dataset?** No.
- **What (other) tasks could the dataset be used for?** Interpretable traffic forecasting, incident classification, incident duration prediction, and traffic causal analysis.
- **Is there anything about the composition of the dataset or the way it was collected and preprocessed/cleaned/labeled that might impact future uses?** The adjacency matrix is generated based on a threshold. It could be revised based on the task requirement.
- **Are there tasks for which the dataset should not be used?** No.

A.1.6 Distribution

- **Will the dataset be distributed to third parties outside of the entity (e.g., company, institution, organization) on behalf of which the dataset was created?** Yes.
- **How will the dataset will be distributed (e.g., tarball on website, API, GitHub)?** Kaggle Dataset.
- **When will the dataset be distributed?** June 11, 2024.
- **Will the dataset be distributed under a copyright or other intellectual property (IP) license, and/or under applicable terms of use (ToU)?** Yes. Please see the section 4.
- **Have any third parties imposed IP-based or other restrictions on the data associated with the instances?** Yes. Please see the section 4.

- **Do any export controls or other regulatory restrictions apply to the dataset or to individual instances?** Yes. The use of data is required to satisfy the Caltrans Terms of Use of PeMS.

A.1.7 MAINTENANCE

- **Who will be supporting/hosting/maintaining the dataset?** The first author of the dataset paper.
- **How can the owner/curator/manager of the dataset be contacted?** After accepting, we will release the email of the owner.
- **Is there an erratum?** No.
- **Will the dataset be updated?** Anual. If someone reports the error to us via GitHub, Kaggle or Email, we will check the data and fix the errors.
- **If the dataset relates to people, are there applicable limits on the retention of the data associated with the instances (e.g., were the individuals in question told that their data would be retained for a fixed period of time and then deleted)?** There is no person information included in XTraffic.
- **Will older versions of the dataset continue to be supported/hosted/maintained?** Yes. The largest difference between the old version and the new version is the time, and it's not hard to maintain the old versions. we will fix the errors reported.
- **If others want to extend/augment/build on/contribute to the dataset, is there a mechanism for them to do so?** No. We don't have enough resources to verify the external contributions.

A.2 STATEMENT OF RESPONSIBILITY

According to the Ownership section in Caltrans Terms of Use of PeMS[1], we can collect and construct a dataset from the source and distribute it. We collected all of the data before 17/05/2024. More details and the introduction of the dataset can be found in the supplementary material. Our XTraffic is released under a CC BY-NC 4.0 International License[2]. The code for the experiments is released under an MIT License[3]. We claim that we are responsible for the data release and collection.

A.3 DETAILS OF XTRAFFIC

Licence. According to the Ownership section in Caltrans Terms of Use of PEMS, we can collect and construct a dataset from the source and distribute it. We collected all of the data before 17/05/2024. More details and the introduction of the dataset are clarified in the Appendix A.3. Our XTraffic is released under a CC BY-NC 4.0 International License. The code for the experiments is released under a MIT License.

Meta data. The meta data for the XTraffic dataset can be accessed at the `https://github.com/XAITraffic/XTraffic/blob/main/xtraffic-metadata.json`.

[1] https://pems.dot.ca.gov/?view=tou

[2] https://creativecommons.org/licenses/by-nc/4.0

[3] https://opensource.org/licenses/MIT

Incidents. The details of incident data features are shown in Table 1.

Table 1: Meta Feature Introduction

Feature	Type	Description
Incident ID	Integer	Unique identifier for each recorded traffic incident.
Duration	Integer	Length of the incident measured in minutes from start to resolution.
Abs PM	Float	Point of the incident in absolute postmile notation along the road.
Fwy	String	The freeway name where the incident occurred.
AREA	String	The city or town where the incident took place.
DESCRIPTION	String	A brief narrative describing the specifics of the incident.
LOCATION	String	The exact address on the freeway where the incident happened.
Type	String	Category of the incident, such as accident, hazard, or road closure.
dt	DateTime	Timestamp indicating when the incident was first reported.

Lane meta features. The details of the lane meta features are in Table 2,

Table 2: Meta Feature Introduction

Feature	Type	Description
Sensor ID	String	Unique identifier for each traffic sensor.
Inner Shoulder Width	Float	Width in meters of the inner shoulder on the lane.
Outer Shoulder Width	Float	Width in meters of the outer shoulder on the lane.
Functional Class	String	Classification of roads based on the function they provide.
Inner Median Type	String	Type of median on the inner side of the road.
Inner Median Width	Float	Width in meters of the median on the inner side of the road.
Road Width	Float	Total width in meters from one side to the other.
Lane Width	Float	Width in meters of each traffic lane on the road.
Design Speed Limit	Integer	Maximum speed limit designed for the road in kilometers per hour.
Terrain	String	Physical features and shape of a landscape, e.g., flat, mountainous.
Population	String	Type of terrain surrounding the road, e.g., urban, rural.
Barrier	String	Description of any barriers along the road, e.g., guardrail, none.
Surface	String	Road surface type, e.g., asphalt, concrete.
Roadway Use	String	Primary use of the road, e.g., commercial, residential.
Length	Integer	The total length of the lane on the road.
Latitude	Float	Geographical latitude of the road's location.
Longitude	Float	Geographical longitude of the road's location.
Abs PM	Float	Point of measurement in absolute postmile notation along the road.
Direction	String	The direction of the lane, e.g., East, North.
Fwy	String	The name of the freeway where the sensor is located in.
District	Integer	The district ID, e.g..
County	String	The county where the sensor is located in, e.g., Orange, Los Angeles.
City	String	The city where the sensor is located in, e.g., Marina, Oakland.
Sensor Type	String	The sensor cateogry, e.g., radars, magnetometers.
Type	String	The level of the road, e.g., mainline, On Ramp.
HOV	String	Whether it is HOV lane or not

A.4 Construction of Adjacency Matrix

Typically, the adjacency matrix is constructed based on distance (Liu et al., 2024). In order to get the real travel distance, we set up an open-source rooting machine engine (Luxen & Vetter, 2011) based on OpenStreetMap, and calculate the shortest travel distance between two sensors based on the coordinate. One more precise adjacency matrix is constructed based on the direction of the lanes and the coordinates of two sensors A and B.

A.5 Definition of Hub and Fringe Nodes

We calculate the degree of each node using the adjacency matrix described in the paper. Nodes with the 500 highest degrees are classified as hub nodes, and those with the 500 lowest degrees are classified as fringe nodes. We match all incidents with the closest node/sensor and also remove the incident samples that the distance between the incident and its closest sensor is larger than 0.05 mile. We count the incident number of the hub nodes and fringe nodes, respectively.

A.6 Experiment Details on Post-Incident Traffic Forecasting

Baselines. The baselines we selected to do the forecasting experiments are typical models in traffic forecasting domain.

- **LSTM**(Hochreiter & Schmidhuber, 1997): A basic model focusing solely on the temporal relationships within traffic data.
- **ASTGCN**(Guo et al., 2019): Enhances the STGCN by incorporating an attention mechanism to better capture node correlations.
- **DCRNN**(Li et al., 2018): An RNN-based model that utilizes diffusion convolution to model traffic flows.
- **AGCRN**(Bai et al., 2020): An adaptive model that combines RNN architecture with an attention mechanism to focus on spatial correlations.
- **GWNET**(Wu et al., 2019): Utilizes a gated mechanism in a TCN framework to filter out irrelevant information effectively.
- **STGODE**(Fang et al., 2021): Uses ordinary differential equations to dynamically model relationships among traffic nodes.
- **DSTAGNN**(Lan et al., 2022): Designed to dynamically capture changing correlations among traffic sensors.
- **D^{2}STGNN**(Shao et al., 2022): A dual-layer spatial-temporal GNN that addresses hidden correlations in traffic data for forecasting.

Implementation Details. We adhered to the identical experimental settings outlined within the work. We divided all the data into training, validation, and test sets in a 6:2:2 ratio. We set the batch size as 24 for DSTAGNN and 64 for all of other models. The learning rate is set as 0.001. Other hyperparameters of models are set as the same as the original settings. Our baselines follow the optimal settings from their sources. For the batch size in the training set, we used different settings to ensure that the model converges as quickly as possible during training. The batch size settings are mentioned in Section 4.2. For **LSTM**, the hidden layer dimension is set as 64, the last linear layer dimension is set as 512. For **ASTGCN**, the dimension of the attention layer is as 64. For **DCRNN**, the number of RNN layers is set as 2, and the dimension for each RNN layer is 64. For **AGCRN**, the hidden dimension is set as 64 for all cells and the embedding dimension is set as 10. For **GWNET**, the dimension of input and output linear layer are set as 32 and 512, respectively. The

dimension of hidden layers is set as 256. For **STGODE**, the regular hyperparameter α is set as 0.8. The thresholds σ and ϵ of spatial adjacency matrix (AM) are set to 10 and 0.5 respectively, and the threshold ϵ of the semantic AM is set to 0.6. For **DSTAGNN**, the attention dimension is set as 32, and the number of attention heads is set as 3. For **D^{2}STGNN**, the hidden dimension is set as 32.

A.7 Experiment Details on Incident Classification

Baselines. We adopt the following representative time series classification baselines.

- **Decision Tree (DT)**: We tailor the canonical decision tree algorithm for the task, recursively partitioning data based on feature values to create a tree-like model that makes classifications at its leaf nodes.
- **TS2Vec** (Yue et al., 2022): It is a universal framework for learning robust and flexible time series representations using hierarchical contrastive learning over augmented context views, making the classification by a linear classifier.
- **gMLP** (Liu et al., 2021): It is a simple network architecture based solely on MLPs with gating, which performs as well as Transformers in key language and vision applications.
- **Sequencer** (Tatsunami & Taki, 2022): It models long-range dependencies using LSTMs without self-attention layers, which enhances performance by reducing the sequence length and creating spatially meaningful receptive fields.
- **OmniScaleCNN** (Tang et al., 2021): It is a 1D-CNN architecture that utilizes a set of prime number-based kernel sizes to efficiently capture optimal receptive field sizes without scale tuning across diverse time series classification tasks.
- **PatchTST** (Nie et al., 2022): It incorporates patching of time series into subseries-level patches and channel-independence to improve long-term forecasting accuracy based on the Transformer backbone.
- **FormerTime** (Cheng et al., 2023): It employs a hierarchical Transformer-based architecture to learn multi-scale feature maps and introduces an efficient temporal reduction attention mechanism and a context-aware positional encoding generator for multivariate time series classification.

Implementation Details. We randomly sampled 9,000 examples to experiment, 3,000 samples per category. The data is divided into training and testing sets in a 7:3 split.

We set the hyperparameters based on the recommended values in the original method and adjust them around those values, taking the parameter values corresponding to the best results as the final result. Specifically, the hyperparameters for each method are as follows:

For the **Decision Tree**, the minimum number of samples required for a leaf node is set to 1, and for splitting an internal node, it is set to 2. In **TS2Vec**, the pretraining stage has an output dimension of 320 and a hidden dimension of 64. The model is trained for 100 epochs with a batch size of 16, and the linear layer is chosen as the downstream classification module with $1e^{-3}$ learning rate. For **FormerTime**, the model is configured with 3 stages, each having 2 layers with a hidden size of 64. The number of slices per stage is 4, 2, 2, with a stride of 4, 2, 2. The model is trained for 100 epochs with $1e^{-3}$ learning rate. In **PatchTST**, the patch length is set to 16, the stride to 8, the number of encoder layers to 2, the number of heads to 8, and the model dimension to 512. **gMLP** is configured with a patch size of 1, a model dimension of 256, a fully forward network dimension of 512, and a depth of 6. **PatchTST**, **TSSequencer**, **OmniScaleCNN**, and **gMLP** are trained for 100 epochs with a batch size of 128, and the learning rate of them is set as $3e^{-4}$.

More Results. Fig. 1 reports the critical difference diagram as presented in (Demšar, 2006), which compares the mean ranks of the baseline methods on the four datasets (three channel-only and all mixed) in the

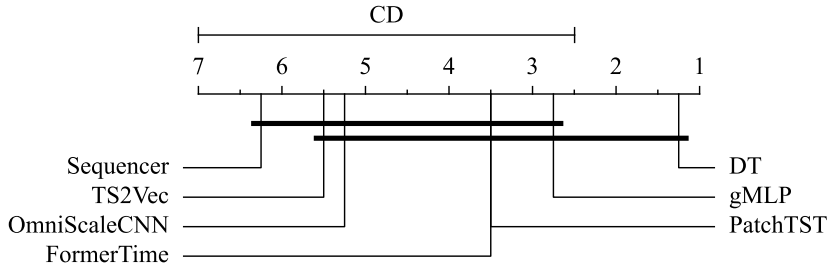


Figure 1: Critical difference diagram over the mean ranks of the compared methods

classification task. The thick horizontal lines in the diagram denote groups of methods whose performance differences are not statistically significant within the critical difference (CD) threshold. It can be seen that DT, gMLP, and PatchTST are among the top-performing methods with the lowest mean ranks, indicating their superior performance. Although DT, gMLP, and PatchTST are highlighted as top performers, the differences among the top five methods are not statistically significant since they are in a group, suggesting comparable effectiveness in this task.

A.8 Experiment Details on Global Causal Analysis

The introduction of MM-DAG. MM-DAG is a score-based causal discovery algorithm. It learns multiple DAGs with multimodal data where their consensus and consistency are maximized. For multimodal data, it proposes a multi-modal regression for linear causal relationship description of different variables by functional principal component analysis. For multitask learning, it uses causal difference to ensure the consistency. The overall optimization problem can be represented as:

$$\begin{aligned}\hat{\mathbf{C}}_{(1)},...,\hat{\mathbf{C}}_{(L)} = \underset{\mathbf{C}_{(1)},...,\mathbf{C}_{(L)}}{\arg\min} & \sum_{l=1}^{L} \frac{1}{2N_l}\|\mathbf{A}_{(l)} - \mathbf{A}_{(l)}\mathbf{C}_{(l)}\|_F^2 \\ & + \rho \sum_{l_1,l_2} s_{l_1,l_2} DCD(\mathbf{W}_{(l_1)}, \mathbf{W}_{(l_2)}) + \lambda \sum_{l=1}^{L} \|\mathbf{C}_{(l)}\|_1 \\ \text{s.t. } & h(\mathbf{W}_{(l)}) = \operatorname{tr}(e^{\mathbf{W}_{(l)}}) - P_l = 0, \forall l\end{aligned}$$

where $\mathbf{A}$ is variables after FPCA, $\mathbf{C}$ are causal matrix and $\mathbf{W}_{(l)ij} = \|\mathbf{C}_{(l)ij}\|_F^2$. s_{l_1,l_2} is the given constant reflecting the similarity between tasks l_1 and l_2, ρ controls the penalty of the difference in causal orders, where larger ρ means less tolerance of difference. λ controls the L_1-norm penalty of causal matrix which guarantees that edges are sparse. In our setting, we set $\lambda = 0.001$, $\rho = 1$ and $s_{l_1,l_2} = 1, \forall l_1, l_2$.

Explaination of the nodes: The details of the nodes in the global causal graph in listed in Table 3.

Constraints of the experiment:In learning DAG, two constraints are placed: (1) edges are not allowed to point toward meta-feature variables since meta-feature variables generally describe environmental and infrastructural contexts that inherently influence other variables rather than being influenced by them. (2) traffic statistics variables are restricted from directing edges towards other nodes since these statistics fundamentally represent outcomes or states of the traffic system, typically influenced by both high-level environmental conditions and specific incidents, rather than serving as direct causes themselves.

Table 3: The description and types of the variables used in traffic causal analysis

Category	Name	Type	Description
high-level variables	Time	Scalar	Day type indicator: $= 1$ if the day is a weekend; $= 0$ otherwise.
	Event	Scalar	Public holiday indicator: $= 1$ if the day is a public holiday; $= 0$ otherwise.
	Visibility	Functional	An integer ranged from 0 to 16 to indicate the visibility of the road in the district.
	Surface	Vector	Attributes describing road material, e.g., concrete, bridge deck.
	Terrain	Vector	Characteristics of the terrain surrounding the road, e.g., flat, rolling.
	Width	Scalar	The width of the road.
	Weather	Functional	An integer ranged from 0 (No rain) to 3 (Heavy rain).
incident variables	Hazard	Functional	Details of any hazards present, e.g., obstacles, spillage.
	NoInj	Functional	Records of accidents with no injuries.
	UnknInj	Functional	Records of accidents with unknown injury statuses.
	1141	Functional	Records of accidents needing an emergency response (coded 1141).
	Fire	Functional	Incidents involving vehicle fires or roadside fires.
	AHazard	Functional	Presence of animals on the road that could cause hazards.
	CarFire	Functional	Specific incidents involving car fires.
traffic statistics	Flow	Functional	Measures of traffic flow, typically in vehicles per hour.
	Occupancy	Functional	Percentage of the road occupied by vehicles at a given time.
	Speed	Functional	Average speed of traffic flow.

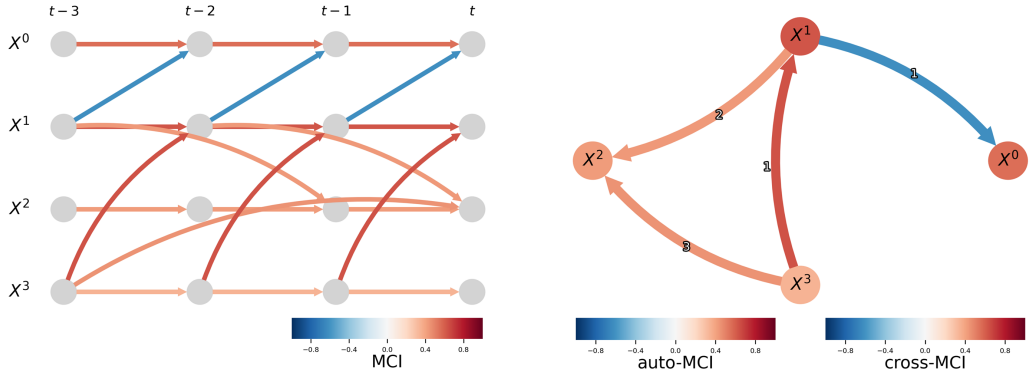
A.9 Experiment Details on Local Causal Analysis

In local causal analysis, We employ the PCMCI$^+$ algorithm to discover the causal relations in traffic data, which utilizes momentary conditional independence (MCI) test to determine the existence of causal links. Typically, the lagged and contemporaneous causal relations are displayed in a dynamic Bayesian network (DBN) as shown in Fig. 2 (a). In this work, to simplify the visualization, we choose to use the process graph as shown in Fig. 2 (b) to aggregate the information in the DBN. In both DBN and process graph, the link color denotes the magnitude of the causal effect measured by the MCI test statistic (e.g., the partial correlation coefficient). The label of a link lists all significant lags of cross-dependencies in process graph. Since we are more interested in the causal links between different traffic nodes, the links denoting auto-dependencies in DBN are summarized into node colors in process graph and the auto-dependency lags are omitted.

The choice of causal structure learning method influences the results of local causal analysis. Ideally, we would like to perform analysis on real cases or datasets with known underlying ground truth of causal dependencies. However, such cases or datasets are rare especially in complex dynamic scenarios such as traffic. To enhance the credibility of the learned causal structure, we use different causal discovery methods and verify the consistency of the results obtained by the different methods. Fig. 3 shows the pre-incident causal graphs of case I learned by score-based method DyNotears (Pamfil et al., 2020) and constrained-based method PCMCI$^+$ (Runge, 2020). The graph structures learned by both methods are similar, but the time lag of the link $X^3 \to X^4$ is different, which is greatly influenced by the sampling frequency of traffic data. Due to the limited number of samples affected by the incidents, we use PCMCI$^+$ to discover post-incident causal structure for its robustness with small sample size and high dimensionality.

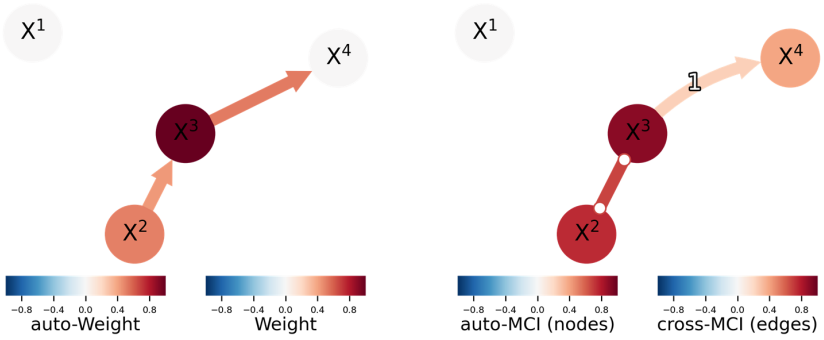
The primary Hyperparameters of $PCMCI^+$ are the maximum time delay $\tau_{\max}$ and the significance threshold for the MCI test α_{PC}. The maximum time delay should be determined based on the specific application, reflecting the maximum expected causal time lag in the scenario under investigation. To identify this maximum time lag, we plot the results of the bivariate lagged conditional independence test. The significance threshold α_{PC} is adjusted on a case-by-case basis to ensure the derived causal structure is reasonable for the analysis. Further details about $PCMCI^+$ implementation and parameter tuning can be found in the public causal discovery tutorials [4]

[4]https://github.com/jakobrunge/tigramite/blob/master/tutorials/causal_discovery/



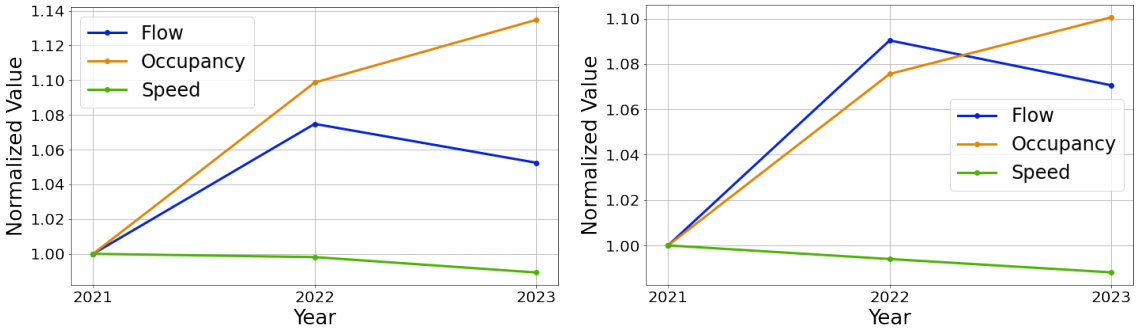
(a) Dynamic Bayesian Network **(b) Process Graph**

Figure 2: Examples of causal graphs for time series variables. (a) Dynamic Bayesian network (DBN). (b) Process graph which summarizes the information in DBN. Process graph aggregates the information in the DBN to simplify the causal structure visualization. While the process graph is nicer to look at, the time series graph better represents the spatio-temporal dependency structure from which causal pathways can be read off. In both graphs, link colors depict the magnitude of the cross-node causal effects as measured by the MCI test statistics. In process graph, node colors depict auto-dependency strength.



(a) DyNotears (Pre-incident) **(b) PCMCI$^+$ (Pre-incident)**

Figure 3: Pre-Incident Causal Graphs for Local Causal Analysis Case Learned by Different Methods. (a) Process Graph Learned by DyNotears. (b) Process Graph Learned by PCMCI$^+$.



(a) Normalized Traffic on Weekday (b) Normalized Traffic on Weekend

Figure 4: (a) and (b) show the year-on-year change trending on Weekday and Weekend, respectively. The time series is normalized for visualization. In 2023, California suffered a long-term natural hazard including blood and storms. Although the traffic flow goes down, the occupancy increases.

A.10 Extend Multi-Year Data Analysis

Currently, we have collected data for January 2021 and 2022 and we conducted two statistical analyses and two case studies.

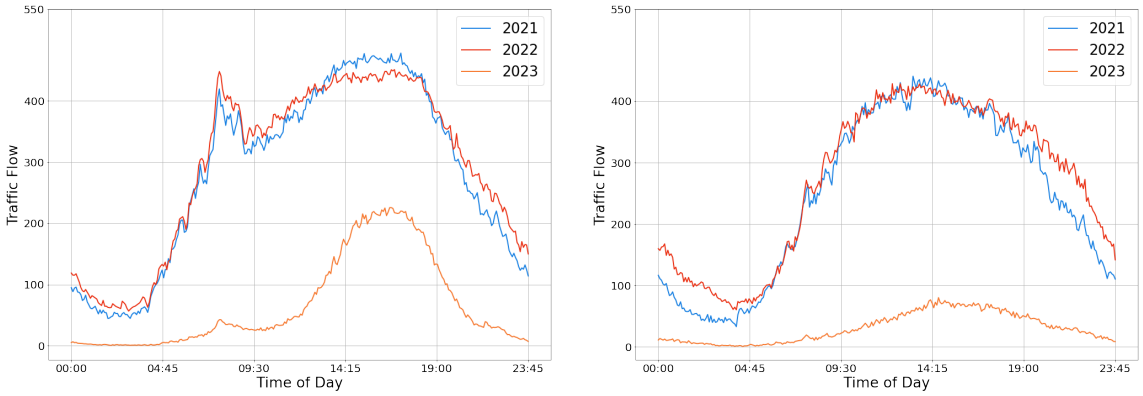
We analyze the weekday and weekend daily trending variation in January based on all sensors excluding those not deployed sensors in 2021. Also, we compared the average flow on the hub road and fringe road during weekdays and weekends.

Year-on-Year Trending on Weekday and Weekday in January. Through year-on-year change analysis and observation, we identified unusual variations in traffic across different years. Due to data limitations, we conducted the analysis only for January data. We divided January traffic data from all sensors into two groups: weekdays and weekends. We then calculated the average traffic for each group, resulting in two distinct traffic change trends. To enhance visualization, we normalized the data by using three types of traffic as baselines and dividing the data from other years by these baselines. The analysis results are illustrated in Fig. 4(a) and (b).

In comparison to 2021, traffic flow increased in 2022, while the average speed decreased in line with the rise in traffic flow. However, in 2023, despite a decrease in traffic flow, traffic speed also declined, which may be attributed to the significant natural disasters in California that persisted until mid-January 2023.

Case Study on Hub Road and Fringe Road. To further explore traffic patterns, we selected a sensor located on a high-traffic segment to observe its daily traffic flow variations. We used the adjacency matrix described in Section 3.2 to compute the degree of each sensor and chose from the top 500 sensors with the highest degrees and no missing data for this analysis. We selected the sensor 717123 as shown in Fig. 5. Similar to the previous analysis, we categorized the traffic flow data from this sensor into weekday and weekend groups. Within each group, we averaged the data across three years. The results, illustrated in Fig. 6, reveal that for weekdays, 2023 still exhibited peak-hour trends. However, for weekends, there is a noticeable decline in traffic flow in 2023 compared to the other two years. This suggests that even under adverse weather conditions, a significant number of people continued to travel to their workplaces.

Figure 5: The location of the selected sensor for the case study. the sensor without missing traffic flow data and with the largest 500 degree among 16,145 sensors.



(a) Traffic flow Variation on Weekday

(b) Traffic flow Variation on Weekend

Figure 6: (a) and (b) show the variation of traffic flow in one day on Weekday and Weekend, respectively. The color represents the traffic flow in different years.

REFERENCES

Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang. Adaptive graph convolutional recurrent network for traffic forecasting. *Advances in neural information processing systems*, 33:17804–17815, 2020.

Mingyue Cheng, Qi Liu, Zhiding Liu, Zhi Li, Yucong Luo, and Enhong Chen. Formertime: Hierarchical multi-scale representations for multivariate time series classification. In *Proceedings of the ACM Web Conference 2023*, pp. 1437–1445, 2023.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine learning research*, 7:1–30, 2006.

Zheng Fang, Qingqing Long, Guojie Song, and Kunqing Xie. Spatial-temporal graph ode networks for traffic flow forecasting. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 364–373, 2021.

Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 922–929, 2019.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Shiyong Lan, Yitong Ma, Weikang Huang, Wenwu Wang, Hongyu Yang, and Pyang Li. Dstagnn: Dynamic spatial-temporal aware graph neural network for traffic flow forecasting. In *International Conference on Machine Learning*, pp. 11906–11917. PMLR, 2022.

Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations (ICLR)*, 2018.

Hanxiao Liu, Zihang Dai, David So, and Quoc V Le. Pay attention to mlps. *Advances in neural information processing systems*, 34:9204–9215, 2021.

Xu Liu, Yutong Xia, Yuxuan Liang, Junfeng Hu, Yiwei Wang, Lei Bai, Chao Huang, Zhenguang Liu, Bryan Hooi, and Roger Zimmermann. Largest: A benchmark dataset for large-scale traffic forecasting. *Advances in Neural Information Processing Systems*, 36, 2024.

Dennis Luxen and Christian Vetter. Real-time routing with openstreetmap data. In *Proceedings of the 19th ACM SIGSPATIAL international conference on advances in geographic information systems*, pp. 513–516, 2011.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations*, 2022.

Roxana Pamfil, Nisara Sriwattanaworachai, Shaan Desai, Philip Pilgerstorfer, Konstantinos Georgatzis, Paul Beaumont, and Bryon Aragam. Dynotears: Structure learning from time-series data. In *International Conference on Artificial Intelligence and Statistics*, pp. 1595–1605. Pmlr, 2020.

Jakob Runge. Discovering contemporaneous and lagged causal relations in autocorrelated nonlinear time series datasets. In *Conference on Uncertainty in Artificial Intelligence*, pp. 1388–1397. Pmlr, 2020.

Zezhi Shao, Zhao Zhang, Wei Wei, Fei Wang, Yongjun Xu, Xin Cao, and Christian S. Jensen. Decoupled dynamic spatial-temporal graph neural network for traffic forecasting. *Proc. VLDB Endow.*, 15(11): 2733–2746, sep 2022. ISSN 2150-8097.

Wensi Tang, Guodong Long, Lu Liu, Tianyi Zhou, Michael Blumenstein, and Jing Jiang. Omni-scale cnns: a simple and effective kernel size configuration for time series classification. In *International Conference on Learning Representations*, 2021.

Yuki Tatsunami and Masato Taki. Sequencer: Deep lstm for image classification. *Advances in Neural Information Processing Systems*, 35:38204–38217, 2022.

Z Wu, S Pan, G Long, J Jiang, and C Zhang. Graph wavenet for deep spatial-temporal graph modeling. In *The 28th International Joint Conference on Artificial Intelligence (IJCAI)*, 2019.

Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. Ts2vec: Towards universal representation of time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 8980–8987, 2022.