teLLMe Why (Ain't Nothing but a Jam): Exploratory Causal Analysis of Urban Driving Data

Qiwei Li **Jorge Ortiz**
Rutgers University, Department of Electrical and Computer Engineering
{qiwei.li, jorge.ortiz}@rutgers.edu

Abstract

Traffic agencies now have access to large volumes of video-derived data for studying safety and congestion. Most of these data are observational and collected without interventions, which makes causal questions such as "How would rain change traffic density?" difficult to answer. We present teLLMe, a system for exploratory causal analysis of urban driving datasets. The system starts from a structured event table built from dashcam annotations and combines causal structure learning with the PC algorithm, bootstrap-based stability checks, and query-specific effect estimation using linear regression and DoWhy. Natural-language questions are mapped to structured causal queries through a schema-aware LLM, enabling users to specify treatments, outcomes, and subpopulations. teLLMe returns a "Causal Card" that summarizes effect estimates, adjustment sets, DAG support, and assumptions, followed by a short natural-language explanation. Case studies on BDD-derived traffic events show that the system can surface plausible relationships involving weather, peak hours, and traffic density, while making uncertainty and modeling choices explicit. The system is designed as a tool for hypothesis generation and expert reasoning rather than a source of definitive causal claims.

1 Introduction

Cities now generate large amounts of video through dashcams, CCTV, and mobile devices. These streams capture routine traffic behavior and environmental conditions, and they offer a rich source of information for studying safety and congestion. Analysts often want to ask causal questions about this data. For example, does rain reduce traffic density at urban intersections, or do peak-hour periods increase demand on specific corridors. Most video, however, is observational and collected without interventions, which makes it difficult to distinguish genuine causal effects from confounding and exposure differences.

Turning raw video into variables that support causal reasoning introduces several challenges. The underlying data are imbalanced and confounded: weather, time of day, scene type, and demand patterns interact in ways that are not controlled by design, and some combinations occur rarely. The analysis pipeline itself is multi-stage. Object detections must be aggregated into event-level features, a plausible causal structure among those variables must be inferred, and treatment effects must be estimated for specific questions. Any resulting estimates must also be communicated to practitioners who are not expected to interpret causal graphs or adjustment rules.

We introduce teLLMe, a system that connects these stages into a single workflow for exploratory causal analysis of urban driving data. The overall architecture of the system is shown in Figure 1. teLLMe starts from a structured event table built from dashcam annotations and learns a causal graph over a curated set of variables using the PC algorithm with domain constraints and bootstrap resampling. Users pose questions in natural language. A schema-aware LLM translates each question into a formal causal query specifying treatments, outcomes, and subpopulation filters. teLLMe then selects a backdoor adjustment set based on the learned graph and estimates average treatment effects

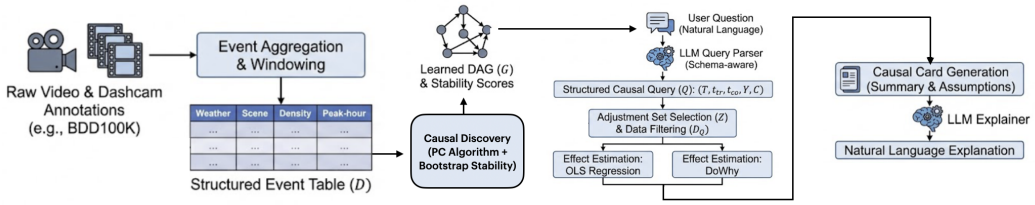

Figure 1: teLLMe has an offline phase that aggregates dashcam annotations into fixed-length windows, constructs a structured event table, and learns a causal graph using the PC algorithm with domain constraints and bootstrap stability. Online, a schema-aware LLM converts natural-language questions into structured causal queries specifying treatment, control, outcome, and filters. The system selects a backdoor adjustment set from the learned DAG, filters the event table, and estimates effects with linear regression and DoWhy. It returns a Causal Card with effect estimates, adjustment sets, DAG evidence, and assumptions, followed by a brief explanation.

using linear regression and DoWhy. The results are presented in a Causal Card that reports effect estimates, adjustment sets, graph evidence, and key assumptions, followed by a brief natural-language explanation.

The contribution of teLLMe is the integration of these components into a query-driven system that makes causal assumptions explicit and keeps the workflow reproducible. The goal is to support the generation and inspection of plausible causal hypotheses from large video-derived datasets and to give domain experts a structured way to interrogate the evidence.

2 Data and System Overview

2.1 Event dataset from dashcam annotations

We work with a structured event table built from a large dashcam corpus such as BDD100K [Xu et al., 2017]. Raw detections and metadata are aggregated into fixed-length windows, each represented by a row with weather labels, scene type, traffic density, peak-hour status, and basic temporal indicators (e.g., weekday/weekend and time-of-day bins). Traffic density is computed as the count of detected vehicles per minute. We drop windows with missing labels and clip extreme values to avoid outliers. Because some combinations of weather and scene type are rare, we create a balanced subset using stratified sampling for the main analyses and retain the full dataset for sensitivity checks.

2.2 System architecture

Figure 1 provides an overview of the system architecture. teLLMe is organized around two linked stages: an offline discovery phase that builds the structural substrate for causal reasoning, and an online query phase that turns user questions into structured causal analyses. The offline stage prepares the data and learns a causal graph with stability information, and the online stage parses user queries, selects adjustment sets, and estimates effects, producing a Causal Card for each query.

Offline. We standardize variables, learn a causal graph over selected features using the PC algorithm [Spirtes et al., 2000, Kalisch and Bühlmann, 2007], apply domain constraints to rule out implausible directions, and run bootstrap resampling to record edge stability. The resulting DAG and stability scores are stored for later use.

Online. Users provide a natural-language question. A schema-aware LLM converts it into a structured causal query with a treatment, outcome, and optional filters. The system selects a backdoor adjustment set based on the learned DAG, filters the event table accordingly, and estimates an average treatment effect using linear regression and DoWhy [Sharma and Kiciman, 2020]. The output is a Causal Card that reports effect estimates, adjustment sets, graph evidence, and assumptions, along with a short natural-language explanation.

3 Methods

3.1 Causal graph learning with PC and bootstrap

Let V denote a subset of event-level variables deemed relevant for causal analysis, such as

$$V = \{\texttt{weather}, \texttt{is\_peak\_hour}, \texttt{traffic\_density}, \texttt{scene\_type}, \texttt{weekday}, \texttt{time\_bin}\}.$$

We treat the event table as samples from a joint distribution over V and aim to learn a directed acyclic graph (DAG) G that encodes candidate causal relationships among these variables.

We use the PC algorithm [Spirtes et al., 2000, Kalisch and Bühlmann, 2007] as implemented in `pgmpy`. PC starts from a complete undirected graph over V and iteratively removes edges based on conditional independence tests, then orients the remaining edges using a set of logical rules. We incorporate domain knowledge through *forbidden edges* (e.g., disallowing edges from `traffic_density` to `weather`) and, if desired, *required edges*. All variables used for PC are discretized or encoded as needed for the chosen independence tests.

To quantify robustness, we perform a simple bootstrap stability analysis [Spirtes et al., 2000, Kalisch and Bühlmann, 2007]. We draw $B = 20$ bootstrap resamples of the event table (with replacement), run PC on each resample, and record how often each directed edge appears. This yields a stability score $s(e) \in [0, 1]$ for each edge e in G, which we later expose in the Causal Cards. Intuitively, edges with high $s(e)$ are more stable under resampling, while edges with low $s(e)$ should be treated with caution.

3.2 Query parsing and DAG-based adjustment selection

Users interact with teLLMe through natural-language questions. To bridge between free text and the fixed dataset schema, we define a structured *Causal Query*:

$$Q = (T, t^{\text{treated}}, t^{\text{control}}, Y, C),$$

where T is the treatment variable, t^{treated} and t^{control} are two values of T, Y is the outcome variable, and C is a set of conditioning constraints (e.g., `scene_type` = intersection, `is_peak_hour` = 1).

We prompt an LLM with (i) the dataset schema, including variable names, types, and allowed categorical values, and (ii) the user question, and instruct it to output Q in a constrained JSON format. The resulting specification is then validated against the schema: we enforce strict type checking, reject references to undefined variables, and discard invalid categorical values. If validation fails, the system returns a user-facing error and optionally falls back to a simple rule-based parser for a subset of templates. This schema-aware parsing layer is crucial for preventing hallucinated columns and keeping causal queries grounded in the actual dataset.

Given Q and the learned DAG G, we select a backdoor adjustment set Z for the treatment–outcome pair (T, Y). In the simplest implementation, we start from a candidate list of covariates (e.g., `scene_type`, `weekday`, `time_bin`) and include those that are parents of T or Y in G while avoiding descendants of T. This heuristic approximates a backdoor adjustment set [Pearl, 2009] and encodes our modeling choice about which variables to condition on when estimating the effect of T on Y.

3.3 Effect estimation and Causal Cards

For the filtered dataset $\mathcal{D}_Q$, we estimate an average treatment effect

$$\tau_Q = \mathbb{E}[Y \mid do(T = t^{\text{treated}}), C] - \mathbb{E}[Y \mid do(T = t^{\text{control}}), C],$$

using the assumptions encoded in G and the selected adjustment set Z.

Linear regression (OLS). We fit an ordinary least squares model

$$Y = \alpha + \beta T + \gamma^\top Z + \varepsilon,$$

with categorical variables in Z one-hot encoded. The coefficient β is taken as the estimate of τ_Q, and we report its point estimate, standard error, and confidence interval.

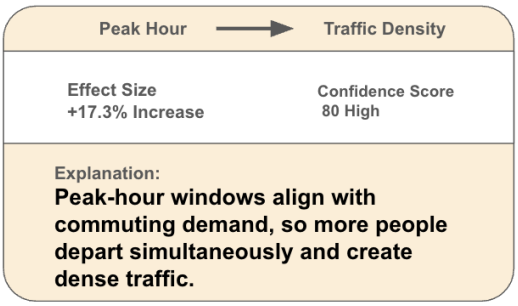

Figure 2: Example Causal Card summarizing the query, effect estimate, adjustment set, DAG information, and a short explanation.

Estimator	ATE	95% CI	N (treated/control)
OLS	0.024	[0.014, 0.034]	4,268 / 6,154

is_peak_hour	count	mean	std	p25	median	p75	min	max
0	6154	0.3827	0.2466	0.20	0.35	0.50	0	1
1	4268	0.5291	0.2835	0.30	0.50	0.75	0	1

Figure 3: ATE summary (top) and outcome distribution (bottom) for peak-hour vs. off-peak on highway traffic density.

DoWhy backdoor estimator. We also construct a DoWhy [Sharma and Kiciman, 2020] *CausalModel* using $\mathcal{D}_Q$, the treatment T, the outcome Y, and a DOT version of the learned DAG G. DoWhy identifies a backdoor estimand and estimates it via linear regression, giving a second estimate of τ_Q under the same adjustment set. DoWhy supports additional estimators (e.g., inverse-propensity and doubly robust methods), but we use linear regression for consistency with the OLS baseline.

Optionally, we bootstrap $\mathcal{D}_Q$ to examine the spread of the estimated effect for the chosen estimator. This gives an empirical measure of stability beyond the usual standard errors.

Causal Cards. Each query is summarized in a compact Causal Card with the parsed variables, the effect estimate and confidence interval, treated/control sample sizes, any direct $T \to Y$ edge and its stability, and relevant caveats. A brief natural-language explanation is generated from these fields.

A short natural-language explanation is then generated from the card for non-technical users.

4 Case Studies

We illustrate teLLMe on several representative queries using the BDD-derived event dataset. All estimates and confidence intervals come from the specified subpopulations and adjustment sets.

4.1 Weather and traffic density at urban intersections

> *What is the effect of rainy versus clear weather on traffic density at urban intersections during peak hours?*

For this query, the treatment is `weather`, with `rainy` as the treated condition and `clear` as the control. The outcome is `traffic_density`, restricted to windows where `scene_type=urban_intersection` and `is_peak_hour=1`. The adjustment module selects `time_of_day` and `total_objects`. In this run, the learned DAG does not contain a direct `weather` $\to$ `traffic_density` edge.

Using OLS with the selected adjustment set, the estimated effect is -0.036 with a 95% confidence interval from -0.047 to -0.024, based on 1,840 rainy windows and 7,681 clear windows.

4.2 Peak-hour effects on highway traffic density

How do peak-hour periods affect traffic density on highways under clear weather?

In this query, the treatment is `is_peak_hour` and the outcome is `traffic_density`. The conditions restrict the data to `scene_type=highway` and `weather=clear`. The adjustment module selects `time_of_day`, `weather`, and `total_objects`. The learned DAG for this subpopulation does not contain a direct `is_peak_hour` $\rightarrow$ `traffic_density` edge.

Using OLS with the selected adjustment set, the estimated effect is 0.024 with a 95% confidence interval from 0.014 to 0.034, based on 4,268 peak-hour windows and 6,154 off-peak windows.

4.3 Sensitivity to adjustment and balancing choices

We compare DAG-based adjustment with a fixed adjustment set and repeat analyses on the imbalanced dataset. Ignoring the DAG sometimes produces larger effects and narrower intervals, consistent with under-adjustment. Analyses on the imbalanced data occasionally produce more extreme estimates and greater uncertainty. These differences highlight the influence of adjustment and sampling choices; teLLMe surfaces these decisions directly in the Causal Cards.

5 Discussion and Limitations

teLLMe works entirely with observational event data, so all effects depend on the assumptions encoded in the learned DAG and the selected adjustment sets. Several important factors—driver intent, road surface conditions, and weather severity— are not observable in dashcam footage, which means the reported effects should be treated as plausible explanations rather than definitive causal claims.

The system treats event windows as independent and does not model temporal or spatial structure. Traffic patterns often depend on both, and capturing those dependencies would require different causal discovery methods and richer data. Measurement limits also constrain the questions we can answer. Traffic density is straightforward to compute, but more safety-oriented surrogates, such as near-miss measures, are harder to derive reliably from video alone. We have not yet evaluated the Causal Cards with practitioners. They are designed to make assumptions and uncertainty clear, but we do not yet know how analysts or planners interpret them or what forms of guidance they find most useful.

Beyond this standalone prototype, teLLMe is being developed as part of the Redddot project, a broader platform for participatory urban safety and mobility analytics. Redddot seeks to give planners, researchers, and community stakeholders access to interpretable views of heterogeneous urban data, including video-derived events, traffic indicators, and contextual information about places. Within this context, teLLMe plays the role of a causal reasoning and explanation module: it turns dashcam-derived event tables into queryable "what-if" analyses, and its Causal Cards can be surfaced alongside other Redddot views to show how candidate effects, uncertainty, and assumptions relate to specific locations and populations. This connection grounds teLLMe's design in a concrete application setting and highlights its potential to mediate human–AI collaboration around urban decisions.

The pipeline gives analysts and other Redddot stakeholders a direct way to test specific causal questions on video-derived data while keeping assumptions explicit. The resulting estimates are intended as inputs to further analysis and deliberation, not final answers.

Acknowledgements

This work was supported by the National Science Foundation as part of the Center for Smart Streetscapes under Cooperative Agreement EEC-2133516 and by NSF Grant No. 2429672.

References

Markus Kalisch and Peter Bühlmann. Estimating high-dimensional directed acyclic graphs with the pc-algorithm. *Journal of Machine Learning Research*, 8(Mar):613–636, 2007.

Judea Pearl. *Causality*. Cambridge university press, 2009.

Amit Sharma and Emre Kiciman. Dowhy: An end-to-end library for causal inference. *arXiv preprint arXiv:2011.04216*, 2020.

Peter Spirtes, Clark N Glymour, and Richard Scheines. *Causation, prediction, and search*. MIT press, 2000.

Huazhe Xu, Yang Gao, Fisher Yu, and Trevor Darrell. End-to-end learning of driving models from large-scale video datasets. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.