# Recurrent Road Bottleneck Detection via Causal Origin-Victim Attribution

Om Jain
*Lepton Software*
Gurugram, India
omjainoj12@gmail.com

Umang Saraf
*Lepton Software*
Gurugram, India
umang.saraf@leptonsoftware.com

***Abstract*—Recurring road bottlenecks, locations that jam on a regular daily or weekly schedule, account for most urban congestion, yet identifying the few places that *cause* it is hard: on a network-wide traffic feed, most slow roads are not bottlenecks but spillback from a jam elsewhere. We present a fully automated, city-general pipeline that turns a coarse (10-minute) per-segment traffic feed into a ranked list of recurring bottleneck sites. Its core step classifies every congested road as a causal ORIGIN or a spillback VICTIM by a bounded downstream walk that applies the onset-ordering principle of congestion propagation on a road graph built from geometry alone. Each origin is scored by a confidence that gates a logistic term with a causal necessity condition (a queue head must have somewhere to drain), and per-city signal calibration is self-bootstrapping, requiring no external labels, so the same model transfers across cities. Deployed for Hyderabad over 30 days, the method finds that $83.6\,\%$ of congested road-segments are spillback and only $16.4\,\%$ are causal origins, reducing a naive slow-road map by $6.1\times$ to $124$ recurring bottleneck sites; the same pipeline runs unchanged on additional cities, and is integrated into a deployed traffic-analytics product.**

***Index Terms*—traffic bottleneck detection, recurrent congestion, real-time traffic data, congestion propagation, intelligent transportation systems**

## I. Introduction

Urban road congestion is dominated not by one-off incidents but by *recurring* bottlenecks: specific locations that jam on a regular daily or weekly schedule. For a city traffic authority the operational question is concrete: given a citywide, real-time view of how fast every road is moving, *which few places actually cause the recurring congestion*, so that limited engineering and enforcement effort can be aimed at them? The desired output is not a heatmap but a short, ranked list of causal sites, each with when it jams, how badly, and how far its queue reaches.

Classic bottleneck identification established both the concept and a detection rule: an *active bottleneck* discharges an upstream queue while traffic immediately downstream stays free, and can be found from a speed differential between adjacent sensors [1]–[3]. That line of work, however, assumes *dense fixed loop detectors* along instrumented freeways. A modern city instead has a network-wide feed of per-segment speeds at a coarse (e.g. 10-minute) cadence covering ordinary streets. Applied to such a feed, the active-bottleneck idea degrades into a naive speed-threshold map that lights up every slow road, but most slow roads are not bottlenecks. They are *spillback*: slow only because a jam somewhere downstream has backed up into them.

The core technical challenge is therefore causal attribution, and it is not resolved by simply thresholding speed. One must separate the *queue head* that causes a jam from the roads that inherit its queue, and do so (i) on a coarse, network-wide feed rather than dense freeway sensors; (ii) automatically and at city scale; (iii) for patterns that *recur* across many days, not single snapshots; and (iv) across different cities without hand-retuning. Congestion-propagation research supplies the guiding principle: congestion appears at the bottleneck first and travels upstream, so temporal precedence orders cause before spillback [4]–[6]. Such work is typically demonstrated on dense trajectory or loop grids and is not tied to automated, recurrent, per-city bottleneck reporting.

We present a fully automated pipeline that produces exactly this list from a network-wide 10-minute traffic feed (Fig. 1). It reduces each road to a median speed ratio, extracts congestion episodes, and, as its core step, classifies every congested road as an ORIGIN (a causal queue head) or a VICTIM (spillback) by a bounded downstream walk that applies the onset-ordering principle on the road graph. Each origin is scored by a confidence that multiplies a logistic term by a *necessity gate* encoding a causal precondition (a true queue head must have somewhere downstream to drain), and the raw signals are normalized per city by a self-bootstrapping calibration that needs no external reference, so the same model transfers across cities. Single-day origins are aggregated into recurring sites with a recurrence class and a severity ranking.

Deployed for one city (Hyderabad) over 30 days, the method finds that only $16.4\,\%$ of congested road-segments are causal origins while $83.6\,\%$ are spillback; it thus reduces a naive slow-road map by $6.1\times$ to $124$ ranked recurring bottleneck sites, and the same pipeline runs unchanged on other cities. Our contributions are:

- a fully automated, *city-general* pipeline that turns a coarse network-wide traffic feed into a ranked list of recurring bottleneck sites, with no per-city code (Sections IV-V);
- a causal ORIGIN/VICTIM attribution that isolates queue heads from spillback by onset-ordering on the road graph, quantified by the $83.6\,\%$ spillback result (Sections IV-D, V-B);

- a *necessity-gated* confidence score and a *self-bootstrapping* per-city calibration that enables cross-city transfer without external labels (Section IV-E); and
- a real deployment with descriptive evidence over a 30-day city window and a qualitative multi-city generality check (Section V).

## II. Related Work

**Bottleneck identification.** The active-bottleneck concept and its speed/occupancy-differential detection were established on freeway loop detectors: Cassidy and Bertini characterize activation from cumulative curves at adjacent detectors [1], and the PeMS line operationalizes scalable identification and delay-based ranking from dense sensor data [2], [3]. These methods define what a bottleneck is and how to rank it, and we adopt their delay-based accounting. They assume dense fixed sensors on instrumented freeways, however, whereas we target a coarse, network-wide street feed and must also decide, for every congested segment, whether it is a cause or spillback.

**Congestion propagation and causality.** Kinematic-wave theory [4], [5], [7] explains why congestion forms at a bottleneck and propagates upstream, giving the temporal precedence we rely on; data-mining work reconstructs propagation as directed graphs/trees over links and attributes the source to the earliest-onset link [6]. We adopt this onset-ordering principle rather than claim it, and differ in setting and product: we apply it on a sparse citywide feed and fold it into automated, *recurrent*, per-site reporting with a calibrated per-origin confidence.

**Recurrent congestion.** A long line separates recurrent from non-recurrent congestion and mines its temporal regularity, from loop-detector delay decomposition [8] to periodicity mining on vehicle mobility data [9]. Our recurrence classes build directly on this; we complement it with an explicit *trend* axis (emerging/fading) for sites that are still forming or decaying, which periodicity-only descriptions do not capture.

**Components we build on.** Junction structure is inferred from geometry with clustering, following the map/intersection-inference literature [10], [11]. Per-segment percentile normalization of congestion is standard practice in congestion indices [12], [13], which our empirical-CDF calibration follows; our contribution there is the self-bootstrapping procedure that harvests those distributions by running detection uncalibrated. Finally, our confidence gate is a multiplicative, noisy-OR-style combination of evidence [14], [15] specialized to a causal necessity condition on downstream drainage.

## III. Data and Problem Setting

### A. Input data

The method's single source of truth is a *10-minute traffic observation* for each directed road segment: for every 10-minute interval, the segment's current travel time (equivalently, speed) together with a reference free-flow travel time, obtained from a third-party real-time traffic feed. These observations span the full urban street network at a uniform 10-minute cadence, unlike the dense fixed loop-detector feeds confined to instrumented freeways in classic bottleneck identification [2], [3]. Coverage is not total: a segment with no observation in an interval is treated as missing (Section III-B). The road network is a set of directed segments with geometry, from which the pipeline derives its own topology (Section IV-B). No incident feed, signal timing, or labeled bottleneck inventory is used at any stage.

### B. The median speed ratio (MSR)

All downstream stages operate on a single derived signal. Time is discretized into $B{=}144$ ten-minute buckets aligned to local midnight. For road $r$ and bucket $b$, the *median speed ratio* summarizes the traffic observations $s$ recorded for that segment during the interval:

$$\mathrm{MSR}_{r,b} \;=\; \mathrm{median}_{s\in(r,b)}\,\min\!\Big(\kappa,\; \tfrac{t_s^{\mathrm{ff}}}{t_s^{\mathrm{cur}}}\Big), \qquad \kappa = 1.20. \quad (1)$$

MSR is the ratio of free-flow to current travel time, capped at $\kappa$ so that faster-than-free-flow noise cannot dominate; *lower is worse* (MSR=1 is free-flowing, MSR$\to$0 is standstill). We deliberately use a per-segment relative ratio rather than an absolute speed so the same thresholds apply across roads of different design speeds [12], [13]. A bucket is assigned a congestion *tier* by fixed, city-independent cut-points:

$$\mathrm{tier}(\mathrm{MSR}) = \begin{cases} \text{heavy} & \mathrm{MSR} \le 0.45 \\ \text{moderate} & 0.45 < \mathrm{MSR} \le 0.65 \\ \text{mild} & 0.65 < \mathrm{MSR} \le 0.85 \\ \text{free} & \mathrm{MSR} > 0.85. \end{cases} \quad (2)$$

Tiers are derived at read time and never stored, so a threshold change re-tiers the whole history without recomputation. Buckets with no observations are missing (not zero); an optional imputation step fills a short segment reading free-flow with the mean MSR of reliable longer neighbours reached by a bounded breadth-first walk, which we omit here for space.

### C. City-general configuration

The method is city-agnostic: a city is defined entirely by data (a bounding area, a UTM zone, a timezone, and per-stage parameters) held as a configuration record, not in code. Onboarding a new city adds a record and points the pipeline at that city's segments; no algorithm changes. We instantiate the method for one city in Section V, but every stage below is written against this general setting.

## IV. Method

### A. Overview

Fig. 1 shows the pipeline. From the MSR field (Section III-B) each processing day proceeds: (i) per-road congestion *episodes* (Section IV-C); (ii) an ORIGIN/VICTIM classification that separates the *cause* of a jam from its *spillback* and wires spill-back chains on the road graph (Section IV-D); (iii) a *confidence* score for each ORIGIN, normalized per city and gated by a downstream-necessity test (Section IV-E); (iv) typed *sites* that anchor origins to junctions or open road segments (Section IV-F). A single day yields candidate bottlenecks; a bottleneck worth reporting is one that *recurs*, so a final stage aggregates many days into a *recurrent-site* snapshot with a recurrence class and severity (Section IV-G). One processing day is fully independent, which makes the per-day stages trivially parallel across dates and cities.

### B. Road network and junctions

Detection walks a directed *road graph*: one node per quantized segment endpoint (coincident endpoints share a node) and one edge per segment, with geodesic length and a reverse-twin link for divided carriageways. Junctions are recovered from geometry alone, adapting the standard clustering approach to intersection detection [10], [11]: segment endpoints are clustered with DBSCAN (radius $\approx 14\,\mathrm{m}$), then a finer shared-node connectivity graph re-splits carriageways that the coarse radius glued together, so two cross-streets a few metres apart on a divided road remain distinct junctions. Each junction's arms are typed (T/Y/X, roundabout, merge/split, interchange) from arm count and turning geometry. The junction layer is network infrastructure; its detailed geometry rules are deferred to the appendix. Endpoints are snapped to the nearest junction within $12\,\mathrm{m}$ so detection can attribute a congested segment to a junction.

### C. Congestion episodes

For each road, the MSR series is first smoothed by a centered rolling median (window 3) to remove single-bucket spikes, then reduced to *episodes* by an asymmetric hysteresis automaton. An episode *onsets* on a single *heavy* bucket, *sustains* while the tier stays in {heavy, moderate}, and *ends* after two consecutive non-sustaining buckets; episodes shorter than three buckets ($30\,\mathrm{min}$) are discarded. The asymmetry (easy to start, slow to end) and the minimum length suppress flicker while preserving genuine congestion. Intervals are end-exclusive.

### D. Origin vs. victim detection

This stage answers the paper's central question for every congested cell: is this road where the jam *starts* (an ORIGIN), or is it merely slow because a downstream jam backed up into it (a VICTIM)? We adopt the standard temporal-precedence view of congestion propagation (a bottleneck emits an upstream-moving queue, so the cause congests no later than its spillback [4]–[6]), and realize it on a coarse, network-wide feed as a bounded downstream walk.

*Downstream walk.* For a congested cell $(r, b)$, WALK (Alg. 1) performs a breadth-first search downstream on the road graph, bounded by a travel-time budget ($\leq 60\,\mathrm{s}$ at the city free-flow speed) and by two gap counters that let the search bridge short non-congested stretches: a tight budget for true data gaps and a looser one across *mild* traffic. U-turn and opposite-carriageway moves are blocked so the walk cannot cross to the other side of a divided road. The search returns downstream cells that are themselves in an episode.

*Parent choice.* Among those downstream candidates, ordered nearest-first, PICKPARENT selects the first whose episode onset is no later than this cell's onset plus a

**Algorithm 1** Origin/victim classification for one congested cell. WALK is a travel-time-bounded downstream BFS with gap bridging and U-turn guards.

**Input:** cell $(r, b)$, road graph $G$, active episodes
**Output:** role ORIGIN/VICTIM and parent (if any)
$C \leftarrow \text{WALK}(r, b, G)$ ▷ downstream cells in an episode
sort $C$ by travel time (nearest first)
**for** $c \in C$ **do**
  **if** $\text{onset}(c) \leq \text{onset}(r, b) + 1$ **then**
    **return** (VICTIM, $c$)
  **end if**
**end for**
**return** (ORIGIN, $\varnothing$)

one-bucket tolerance. The tolerance accommodates bucket-quantized, near-simultaneous jams. If such a parent exists, $(r, b)$ is a VICTIM pointing at it; otherwise it is an ORIGIN.

*Spill-back chains and continuity.* Victims are wired into chains by pointing each at the next road on its downstream path; intermediate uncongested roads on the path are marked as connectors so a rendered chain is edge-contiguous. Cycles are broken by promoting the earliest-onset real cell to ORIGIN. Two further passes improve continuity on sparse data: a reconciliation pass re-checks each ORIGIN for a slightly farther downstream cause it initially missed, and a hole-overlay pass bridges brief gaps where a road was congested but never formed a full episode. A role-hysteresis pass absorbs sub-three-bucket role flips. These continuity mechanisms are engineering for the sparse, noisy coverage of a network-wide feed rather than changes to the underlying precedence rule.

### E. Confidence: a necessity-gated logistic

Not every ORIGIN is equally trustworthy. Each is scored by a bounded confidence that multiplies a logistic *sufficiency* term by a *necessity* gate:

$$z = \beta_0 + w_g q_g + w_b q_b + w_s q_s + w_j q_j, \tag{3}$$

$$g = \varepsilon + (1 - \varepsilon)\, \max\big(d_w,\ \rho\, q_g\big), \tag{4}$$

$$\text{conf} = g \cdot \sigma(z), \tag{5}$$

with weights $(\beta_0, w_g, w_b, w_s, w_j) = (-1.4, 1.6, 1.0, 1.0, 0.8)$, $\varepsilon = 0.18$, $\rho = 0.7$. The four sufficiency signals are: $q_g$, the growth rate of the queued (MSR-weighted) length behind the origin; $q_b$, the total queue length spanned; $q_s$, the fraction of its episode the road held the ORIGIN role; and $q_j$, junction pressure (how many independent, well-drained origins meet at the same junction). The gate is driven by $d_w$, *downstream clearance*: whether the road just downstream of the origin is actually free enough to be draining a queue.

The gate encodes a causal *necessity* condition (a true queue head must have somewhere to drain) as a multiplicative cap, a soft-OR (noisy-OR-style) combination of clearance with a discounted growth term, floored at $\varepsilon$ so confidence is attenuated, not zeroed, when downstream evidence is ambiguous [14], [15]. A strongly growing queue can partially open the

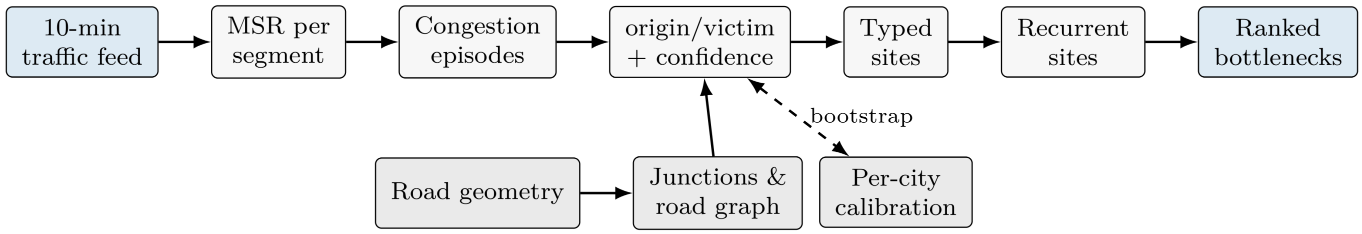


Fig. 1. Pipeline overview. A 10-minute per-segment traffic feed is reduced to a median speed ratio (MSR), segmented into congestion episodes, and classified into causal ORIGINs and spillback VICTIMs scored by confidence; origins become typed sites, aggregated over many days into ranked recurring bottlenecks. The per-day stages (episodes, detection, sites) run independently for each date. Junctions and the road graph are built once from geometry (side input); per-city calibration is self-bootstrapping: detection is run uncalibrated to learn its own normalization.

**Algorithm 2** Self-bootstrapping per-city calibration. Detection is run uncalibrated to harvest the signal distributions it will later be normalized against.

```
Input: city, window [a−Δ+1, a]
Output: per-signal ECDF and scale
for each date d in the window do
    run episodes + detection with calibration off
    collect raw signal values from every ORIGIN
end for
for each signal with ≥ 100 samples do
    build a 101-point ECDF; scale ← 90th percentile
end for
```

gate when the downstream reading is noisy, but unambiguous clearance always dominates. Separating necessity (gate) from sufficiency (logistic) is more faithful than folding clearance into a single weighted sum, where a large queue could mask the absence of any drain.

*Per-city calibration.* The raw signals live on incomparable scales that differ by city. Each is mapped to $[0, 1]$ against the city's own empirical distribution: with calibration, $q$ is the percentile rank of the raw value under a stored 101-point ECDF for that signal; without it, a fixed fallback divisor is used (cold start). Calibration is *self-bootstrapping*: the detector is re-run over a recent window with calibration switched off so it emits raw, unscaled signals, whose distribution then defines the ECDF and a 90th-percentile scale (Alg. 2). This breaks the circularity (the scales the detector needs are learned from the detector run without them) and requires no external reference. Percentile normalization of congestion signals is itself standard [12], [13]; running detection uncalibrated to harvest the distribution is the uncommon step.

### *F. Sites*

Origins are grouped into reportable *sites*. Each ORIGIN is anchored to a *place* (the junction it snaps to, else its road-segment end node) and split into time-contiguous instances (gaps up to two buckets are bridged). Instances shorter than three buckets or with no victims are dropped. A site is typed from its junction shape (T/Y/X, multi-arm, merge/split) or, away from any junction, as a lane drop. For each site we record an impact-weighted confidence and a daily impact

$$\text{day\_impact} \;=\; \sum_{\text{victim cells}} (1 - \text{MSR})\, \ell\, \frac{\Delta}{v_{\text{ff}}}, \tag{6}$$

with segment length $\ell$, bucket width $\Delta$=10 min, and free-flow speed $v_{\text{ff}}$; this is a standard vehicle-hours-of-delay measure [2] summed over the queue. Here $v_{\text{ff}}$ is a single per-city constant; wiring per-road free-flow speeds is future work (Section VI).

### *G. Recurrent sites*

A place that jams once is noise; the reportable output is a *recurrent site*. Over a multi-day window, per-place site-days that are strong enough (confidence $\geq$ 0.50 and $\geq$ 30 min, or a severe rescue) are aggregated. Each place is assigned a recurrence class from its active-day pattern: EVERYDAY, WEEKDAYS, WEEKENDS (structural, by weekday/weekend active fractions with a two-halves stability check), or EMERGING/FADING (a trend axis: a still-forming or decaying pattern). This scheme extends the classic recurrent/non-recurrent distinction [8], [9] with an explicit trend direction. A victim chain is assembled from roads that recur behind the head on a sufficient fraction of days, and active-time windows are cut from the daily delay profile.

All speed and delay statistics derive from one additive *sum-time* primitive over the chain's congested cells, so that adjacent segments and merged corridors combine by summation rather than by re-taking medians. With $F{=}\sum t^{\text{ff}} n$ the total free-flow time, $J{=}\sum t^{\text{ff}} \sum 1/\max(\text{MSR}, \phi)$ the total realized time ($\phi$=0.05 caps the standstill singularity), and $L{=}\sum \ell n$ the total jammed length, the reported jam and free-flow speeds and the per-kilometre delay are algebraic identities of $F$, $J$, $L$:

$$v_{\text{jam}} = 3.6\,\tfrac{L}{J}, \quad v_{\text{ff}} = 3.6\,\tfrac{L}{F}, \quad \text{delay/km} = \tfrac{J-F}{L} \cdot \tfrac{1000}{60}. \tag{7}$$

Additive delay is the standard performance measure [2]; the point here is that expressing every skin as a sum makes the KPIs *mergeable* when corridors are folded together. A site is kept if it clears an OR-floor on delay, distance, or speed loss, and ranked by

$$\text{severity} = \text{frac}_{\max}\,\big(1 + \ln(1 + n_v)\big)\, \text{impact}_{\text{norm}}, \tag{8}$$

where $\text{frac}_{\max}$ is the peak weekday/weekend active fraction, $n_v$ the number of victims, and $\text{impact}_{\text{norm}}$ a normalized blend

TABLE I
FROM ROAD NETWORK TO REPORTABLE BOTTLENECKS (HYDERABAD, 30 DAYS: 1-30 JUNE 2026).

| Stage | Count |
|---|---|
| Road-graph edges (directed) | 9856 |
| Roads congested $\geq 1$ day | 1827 |
| Congested road-buckets (ORIGIN+VICTIM) | 2 602 865 |
| of which ORIGIN (causal) | 427 401 |
| ORIGIN-sites per day (median) | 179 |
| Recurring-bottleneck sites (final) | 124 (clubbed to 117) |

of delay, speed loss, distance, and total impact. The surviving recurrent sites, with class, windows, chain, and severity, are the pipeline's output.

## V. EVALUATION

We evaluate the method as deployed for one city. There is no labeled ground-truth bottleneck inventory for this city (or, to our knowledge, a public one at this granularity); like much of the bottleneck-identification literature [2], [8], we therefore report descriptive results, internal consistency, and a comparison against a naive detector rather than precision/recall against labels (Section VI).

### A. Deployment and data

The method was instantiated for Hyderabad, India, over a $30\,\mathrm{day}$ window (1-30 June 2026). Junction and graph construction over the city's road segments produced a directed *road graph* of 9856 edges. Over the window the per-day stages aggregated 35 171 747 traffic observations into 7 924 164 MSR road-buckets; across all days 1827 distinct roads ($18.5\,\%$ of the network) were congested at least once, and detection produced a median of 179 ORIGIN-sites per day. Table I summarizes this funnel from raw network to reportable output.

### B. Most congestion is spillback

The central empirical result concerns the ORIGIN/VICTIM split. Of the 2 602 865 congested road-buckets in the window, only $16.4\,\%$ (427 401) were classified ORIGIN (causal queue heads); the remaining $83.6\,\%$ (2 175 464) were VICTIM: roads that were slow only because a downstream jam backed up into them. A *naive detector* that reports every congested segment as a bottleneck (the implicit model behind a simple speed-threshold map) therefore over-reports by $6.1\times$ relative to the causal origins, and $83.6\,\%$ of what it would flag as independent bottlenecks are, under our method, attributed to a cause rather than counted separately (Fig. 2). Isolating that one-sixth is the primary function of the detection stage and the main practical value of the system: it turns a wash of slow roads into a short, ranked list of causes. On the retained origins the per-cell confidence has median $0.55$ (10th/90th percentiles $0.21/0.77$), i.e. a usable spread rather than a saturated score.

### C. Recurring bottlenecks

Aggregating the per-day sites over the window yields 124 recurring bottleneck sites (the deployed UI *clubs* co-located

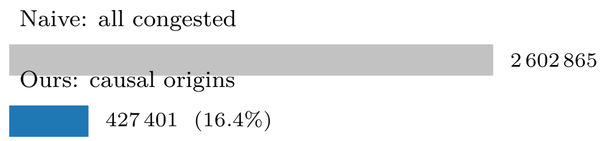


Fig. 2. A naive detector that flags every congested road-segment reports 2 602 865 bottleneck-segments; our ORIGIN/VICTIM attribution keeps only the 427 401 causal origins ($16.4\,\%$), a $6.1\times$ reduction. The remaining $83.6\,\%$ are spillback attributed to a cause.

TABLE II
RECURRING SITES BY RECURRENCE CLASS (HYDERABAD, $n$=124). DELAY, ACTIVE DAYS, AND JAM SPEED ARE PER-CLASS MEDIANS.

| Class | $n$ | Delay (min) | Days | Jam (% ff) |
|---|---|---|---|---|
| Everyday | 73 | 1.7 | 27 | 40 |
| Weekdays | 33 | 1.1 | 11 | 45 |
| Weekends | 13 | 2.7 | 5 | 42 |
| Fading | 4 | 1.1 | 14 | 41 |
| Emerging | 1 | 1.8 | 11 | 44 |

junction arms into 117 display sites). Fig. 4 shows that deployed view: severity-ranked sites with per-site 24-hour delay profiles over a network colored by congestion, summarized as 117 sites and $94.4\,\mathrm{km}$ of typical backup. The sites are dominated by road-segment lane drops (96 of 124) over junction sites (28), and by the *Everyday* recurrence class (73), consistent with structural rather than incidental congestion (Table II). Severity is a unitless, city-relative score (median 0.47, max 3.0). The per-site impact is modest but persistent (Fig. 3): median added delay $1.5\,\mathrm{min}$ (90th percentile $4.9\,\mathrm{min}$), median jam speed $41\,\%$ of free-flow (10th percentile $28\,\%$), and a median queue of $0.6\,\mathrm{km}$ over a median of 22 active days. Table III breaks the sites down by type: junction sites, though fewer, are the more severe, carrying roughly double the median delay of road-segment lane drops.

### D. A representative site

Fig. 5 shows the highest-severity site, a T-junction the system labels after its dominant road. It recurs on all 30 active days (22/22 weekdays, 8/8 weekends), with speed dropping from 37 to $8\,\mathrm{km\,h^{-1}}$ through the site, an added $14.1\,\mathrm{min}$ of delay ($5.8\,\mathrm{min\,km^{-1}}$, $2.4\times$ the city average), and a queue of $2.4\,\mathrm{km}$ spanning 38 roads. The panel's plain-language "why it jams" notes (a large multi-arm crossing, and nearby hospitals and a market) come from an auxiliary POI layer used only for display; they do not feed detection.

### E. City-generality

The method is city-agnostic by construction: a city is a configuration record, and onboarding requires no algorithm change (Section III). Beyond the Hyderabad deployment we ran the same pipeline end-to-end on other cities (including Delhi and Bangalore), confirming that it onboards and produces bottleneck sites with no per-city code change. Those runs used short windows, so we do not report per-city recur-

TABLE III
RECURRING SITES BY TYPE (HYDERABAD, $n$=124). VALUES ARE PER-TYPE MEDIANS; THE FIRST ROW IS A ROAD-SEGMENT TYPE, THE REST ARE JUNCTION TYPES.

| Type | $n$ | Delay (min) | Extent (km) |
|---|---|---|---|
| Road-segment lane drop | 96 | 1.3 | 0.56 |
| T-junction | 8 | 2.4 | 0.66 |
| Merge | 6 | 2.5 | 0.67 |
| Crossroads (X) | 6 | 2.6 | 0.61 |
| Multi-arm | 4 | 2.5 | 0.75 |
| Split | 4 | 1.3 | 0.51 |

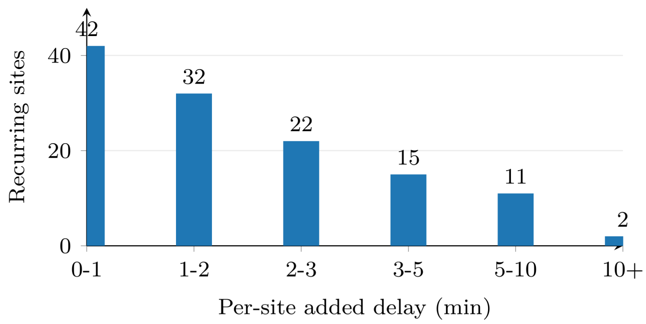


Fig. 3. Distribution of per-site added delay across the 124 recurring sites: most add under 2 min, with a long tail of severe sites exceeding 10 min.

rence statistics here; the point is that generality is structural, not retuned per city.

## VI. LIMITATIONS

Our evaluation has clear boundaries. First, there is *no labeled ground truth*: we report descriptive results, internal consistency, and a naive-detector comparison rather than precision/recall against a validated bottleneck inventory. This is common in the subfield, where such inventories rarely exist at street granularity, but it means we demonstrate plausibility and internal coherence, not verified accuracy. Second, the delay and impact figures assume a *single per-city free-flow speed*; the median-speed-ratio signal and the relative severity ranking are unaffected, but absolute delay-per-kilometre values are approximate, and wiring per-road free-flow speeds is the highest-value accuracy improvement. Third, our *quantitative* results come from a single-city, 30-day deployment; the method is city-general by construction and runs unchanged elsewhere (Section V), but a multi-city quantitative study remains future work. Finally, junction detection and typing are *geometric heuristics*: when a site's type or location looks wrong, the junction layer, not the causal attribution, is usually the cause.

## VII. CONCLUSION

We described a fully automated, city-general pipeline that identifies recurring road bottlenecks from a coarse, network-wide traffic feed, and showed that its central step, separating causal queue heads from spillback by onset-ordering on the road graph, is what makes the output actionable: in a 30-day city deployment, 83.6 % of congested road-segments are spillback, so isolating the 16.4 % that are causes turns a wash of slow roads into a short, ranked list of 124 recurring sites. The necessity-gated confidence and self-bootstrapping per-city calibration let the same model transfer across cities without retuning. The most valuable next steps are wiring per-road free-flow speeds to make absolute delays exact, a multi-city quantitative study, and, where a labeled inventory can be assembled, a formal accuracy evaluation.

## REFERENCES


[1] M. J. Cassidy and R. L. Bertini, "Some traffic features at freeway bottlenecks," *Transportation Research Part B: Methodological*, vol. 33, no. 1, pp. 25–42, 1999.

[2] C. Chen, A. Skabardonis, and P. Varaiya, "Systematic identification of freeway bottlenecks," *Transportation Research Record*, vol. 1867, no. 1, pp. 46–56, 2004.

[3] C. Chen, K. Petty, A. Skabardonis, P. Varaiya, and Z. Jia, "Freeway performance measurement system: Mining loop detector data," *Transportation Research Record*, vol. 1748, no. 1, pp. 96–102, 2001.

[4] M. J. Lighthill and G. B. Whitham, "On kinematic waves II: A theory of traffic flow on long crowded roads," *Proceedings of the Royal Society of London A*, vol. 229, no. 1178, pp. 317–345, 1955.

[5] G. F. Newell, "A simplified theory of kinematic waves in highway traffic, Part I: General theory," *Transportation Research Part B: Methodological*, vol. 27, no. 4, pp. 281–287, 1993.

[6] H. Nguyen, W. Liu, and F. Chen, "Discovering congestion propagation patterns in spatio-temporal traffic data," *IEEE Transactions on Big Data*, vol. 3, no. 2, pp. 169–180, 2017.

[7] P. I. Richards, "Shock waves on the highway," *Operations Research*, vol. 4, no. 1, pp. 42–51, 1956.

[8] A. Skabardonis, P. Varaiya, and K. F. Petty, "Measuring recurrent and nonrecurrent traffic congestion," *Transportation Research Record*, vol. 1856, no. 1, pp. 118–124, 2003.

[9] S. An, H. Yang, J. Wang, N. Cui, and J. Cui, "Mining urban recurrent congestion evolution patterns from GPS-equipped vehicle mobility data," *Information Sciences*, vol. 373, pp. 515–526, 2016.

[10] J. Biagioni and J. Eriksson, "Inferring road maps from global positioning system traces: Survey and comparative evaluation," *Transportation Research Record*, vol. 2291, no. 1, pp. 61–71, 2012.

[11] A. Fathi and J. Krumm, "Detecting road intersections from GPS traces," in *Geographic Information Science (GIScience 2010)*, ser. LNCS, vol. 6292, 2010, pp. 56–69.

[12] F. He, X. Yan, Y. Liu, and L. Ma, "A traffic congestion assessment method for urban road networks based on speed performance index," *Procedia Engineering*, vol. 137, pp. 425–433, 2016.

[13] T. Afrin and N. Yodo, "A survey of road traffic congestion measures towards a sustainable and resilient transportation system," *Sustainability*, vol. 12, no. 11, p. 4660, 2020.

[14] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[15] S. Srinivas, "A generalization of the noisy-or model," in *Proc. 9th Conf. on Uncertainty in Artificial Intelligence (UAI)*, 1993, pp. 208–215.

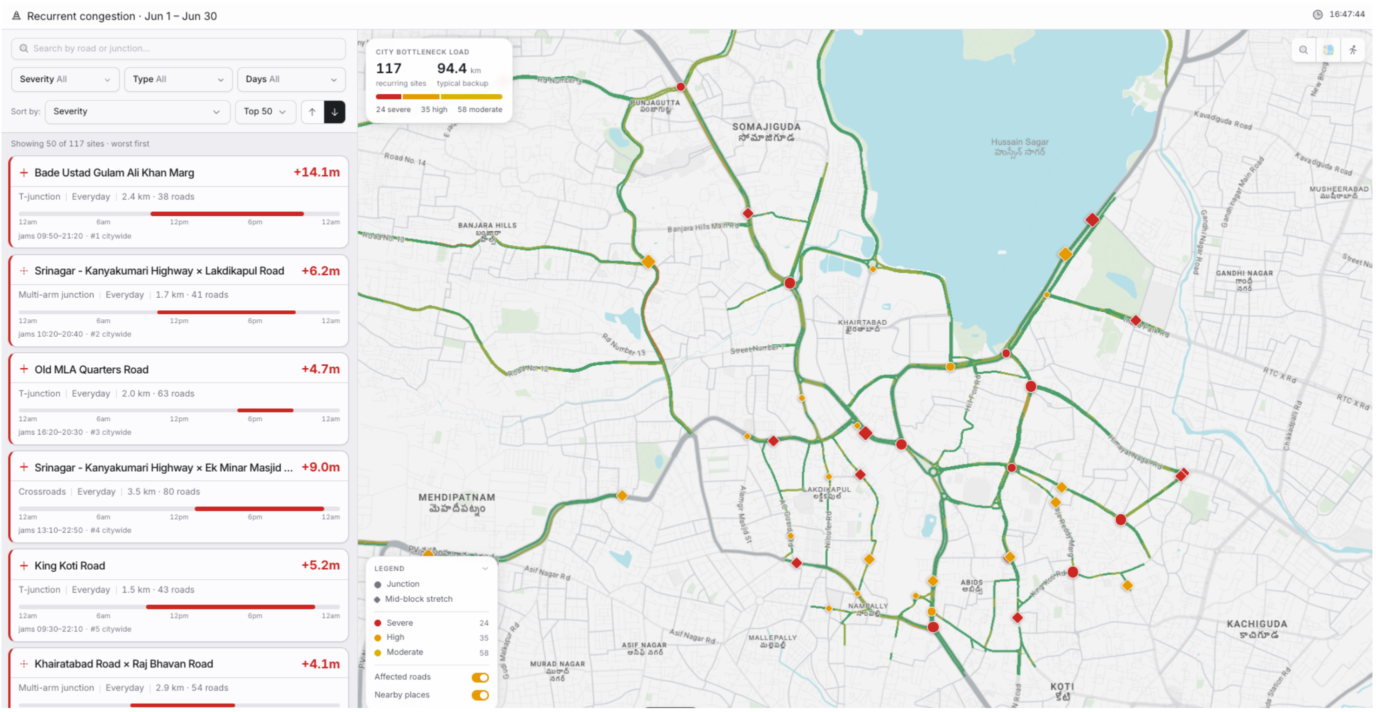


Fig. 4. Deployed recurring-bottleneck view for Hyderabad (30-day window, 1-30 June 2026). The left panel ranks sites by severity with per-site 24-hour delay profiles; the map colors roads by congestion and marks junction (circle) and road-segment (diamond) sites. The summary reports 117 recurring sites and 94.4 km of typical backup.

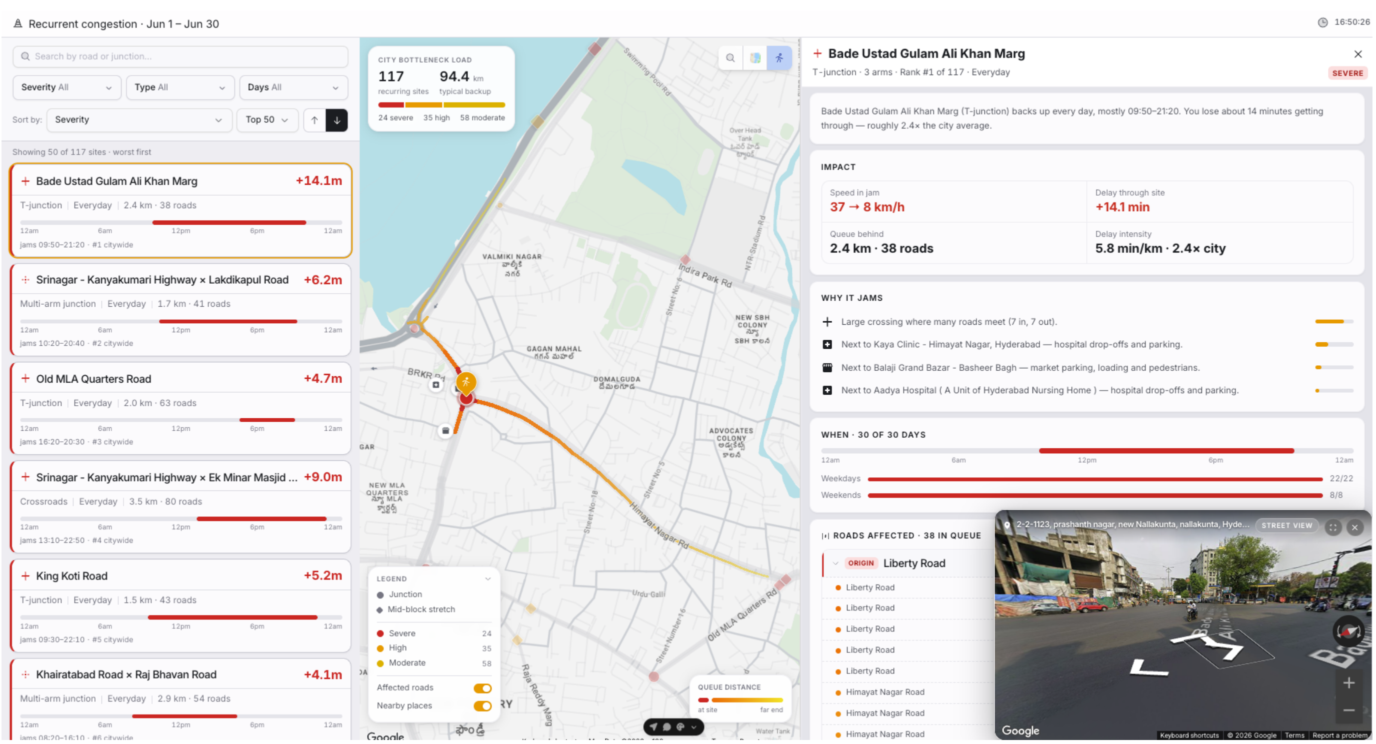


Fig. 5. Site detail for the highest-severity Hyderabad bottleneck: impact KPIs, recurrence pattern (30/30 days), the spillback chain on the map, the roads-affected queue, and a street-level view of the location. Numbers match the database record; the display name is the API's clubbed label.