LEARNING A SCALABLE NEURAL SURROGATE FOR EVENT-LEVEL STATISTICALLY GROUNDED HOTSPOT DETECTION IN SPATIAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

The growing availability of large-scale spatial event data has made hotspot detection increasingly challenging in spatial data analytics, with applications in public health, public safety, and urban analytics. In these domains, statistically grounded methods are widely used but computationally expensive when applied to millions of events on road networks. In this work, we propose a graph neural network framework that learns to approximate a computationally expensive statistical hotspot-detection method on road networks. Specifically, we use a network local K-function-based method to generate supervision, including event-level hotspot labels and associated hotspot radii, and train a graph neural network to jointly predict hotspot membership and radius directly from graph-structured spatial data. We construct heterogeneous graphs of events and road intersections, allowing the model to use both event features and road-network structure. By replacing repeated statistical evaluation at inference time, the learned surrogate accelerates event-level hotspot detection while maintaining good agreement with the statistical detector. We evaluate the surrogate on real-world road networks. On the Los Angeles test graph containing 8 million events, it achieves an F1 score of 0.897 and AP of 0.979. It processes the graph in 0.427 seconds, achieving an approximately 3,400-fold speedup over AH-IBT, the current state-of-the-art method. Radius predictions on the same graph have a mean absolute error of 22.1 meters. The model also transfers zero-shot to unseen cities.

1 INTRODUCTION

Hotspot detection supports decision-making in public health, public safety, and urban planning by identifying regions with elevated event density relative to an expected baseline Xie & Shekhar (2019); Hohl et al. (2020); Shi et al. (2022); Romano & Jiang (2017); Khalid et al. (2018); Qu et al. (2024). For events such as traffic accidents and street crimes, these concentrations are shaped by the underlying road network Liu et al. (2023); Khalid et al. (2018). As event datasets grow, hotspot detection becomes increasingly computationally demanding, with the Chicago City of Chicago and Los Angeles City of Los Angeles datasets reaching 6 million and 16 million events, respectively.

Focusing on network space, where hotspots are constrained to specific infrastructures such as road networks Liu et al. (2023); Wang et al. (2019), hotspot detection can be defined as the task of identifying localized network regions around events, parameterized by a radius in network distance, whose event concentration is statistically significant relative to an expected baseline. Research on hotspot detection techniques is extensive, and existing approaches can be broadly divided into two main categories: clustering-based methods Wang et al. (2019); Deng et al. (2019); Xie et al. (2022), and statistical methods Liu et al. (2023); Xie et al. (2020); Tang et al. (2019). Traditional clustering techniques, such as DBSCAN Ester et al. (1996) and k-means Lloyd (1982), are efficient in clustering events. However, since these techniques do not involve any measure of statistical significance, they are less accurate Xie et al. (2022); Xie & Shekhar (2019) and less preferred by many experts in various domains, e.g., health informatics, urban planning, and public safety. Thus, statistical hotspot-detection methods are often preferred in such critical applications because they explicitly test whether observed clusters are unlikely to arise by chance, reducing spurious detections and improving robustness Xie et al. (2022); Xie & Shekhar (2019).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Although statistical methods use significance tests for examination through scoring functions such as log-likelihood Tang et al. (2017), or density scores Tang et al. (2019), they typically rely on randomization techniques (e.g., Monte Carlo simulation) to assess statistical significance under a null model. While these methods provide stronger statistical evidence, they introduce substantial computational overhead, particularly when applied to large-scale spatial networks. To address these challenges, recent work has explored eliminating expensive randomization while retaining statistical guarantees. In Liu et al. (2023), the authors have eliminated some of the costs associated with Monte Carlo simulation. However, their approach still incurs considerable computational overhead because it requires exhaustive scanning of event-centered regions and implicit enumeration of hotspot radii over large spatial networks. This limitation motivates the principle that computationally expensive methods can be approximated through learning. Existing learning-based approaches primarily formulate hotspot detection as a predictive or classification problem, where hotspots emerge implicitly from predicted node-level scores. In contrast, hotspot detection in spatial statistics requires identifying regions that are statistically significant, and none of these methods are designed to approximate statistically grounded hotspot detection or support scalable hotspot analysis over large event-mapped spatial networks Ying et al. (2018); Hu et al. (2020); Zhang & Cheng (2020); Wang et al. (2025); Sui et al. (2024); Kong et al. (2024); Khetarpaul & Sharan (2026).

We seek to learn a surrogate model of a statistical hotspot detection procedure that maps graph-structured spatial data to event-level hotspot labels and associated spatial extents (radii). In this setting, we first use a statistically grounded hotspot detection method to generate reliable supervision data, and then we train a learning-based model to approximate this process efficiently on large datasets, avoiding repeated statistical evaluation during inference. Can a learned surrogate jointly approximate statistical hotspot membership and radius for millions of events while transferring across cities?

Learning an efficient surrogate for this task presents three related challenges. First, a representation must capture network connectivity and event distributions while remaining compact enough to process millions of events. Second, hotspot membership and radius depend on how the event pattern changes across network neighborhoods of different sizes. The surrogate must approximate both outputs without repeating the statistical method's radius search. Third, differences in road-network structure and event density can alter these patterns across cities. A model trained in one city must therefore retain agreement with the statistical method in unseen cities without city-specific retraining.

The structure of spatial networks makes graph-based modeling particularly well-suited for this problem. Unlike Euclidean domains, distances and neighborhoods are defined along the network topology, and hotspot regions correspond to localized neighborhoods that expand over network paths. This results in irregular, non-grid spatial structure where multi-hop relationships are essential. Consequently, capturing hotspot patterns requires modeling interactions between events and their surrounding network context, which cannot be naturally handled by grid-based models or standard density estimation techniques. Graph neural networks provide a natural framework by enabling message passing over network-defined neighborhoods.

Motivated by these observations, we aim to leverage graph machine learning to develop a model that approximates statistically grounded hotspots with lower computational cost and improved scalability. Our main contributions are summarized as follows:

- **Graph construction for spatial network hotspot detection.** We construct a compact heterogeneous graph that retains individual events and connects them through road intersections. Event-related edges grow linearly with the number of events, while shared feature definitions encode local geometry and relative event density across cities.
- **Learning-based approximation of statistical hotspot detection.** We propose a graph learning framework that approximates a statistical event-level hotspot detection procedure using supervision derived from a network local K-function–based method, jointly predicting hotspot membership and associated spatial extents (radii) for fine-grained characterization of hotspot regions.
- **Scalable and generalizable hotspot detection.** We demonstrate effectiveness on real-world datasets with up to 8 million events per graph, achieving an F1 score of 0.897 on the 8-million-event LA test graph with 0.427 seconds of GPU forward time, and show that the learned model generalizes across regions and cities with F1 scores between 0.884 and 0.939 on the three unseen cities.

2 Related Work

We consider hotspots on spatial networks as regions with event concentrations significantly higher than expected under a specified null model. Prior work on hotspot detection in spatial networks broadly falls into (i) *clustering-based* methods Wang et al. (2019); Deng et al. (2019); Xie et al. (2022); Ester et al. (1996); Lloyd (1982) and (ii) *statistical* methods Liu et al. (2023); Xie et al. (2020); Tang et al. (2017; 2019). The key distinction is whether spatial concentrations are identified from proximity or density alone, or assessed through explicit significance testing. In this paper we use *statistically grounded* to describe the latter criterion.

Clustering-based hotspot detection. Clustering-based approaches group events according to proximity or density in network space. Relevant approaches include partition-based clustering, DBSCAN-style methods, and hierarchical clustering, together with their adaptations to network-constrained settings Ester et al. (1996); Lloyd (1982); Wang et al. (2019); Deng et al. (2019); Xie et al. (2022). These methods are typically computationally efficient and scale well. However, without an explicit significance test, a detected cluster does not by itself establish a statistically significant departure from a null model Xie & Shekhar (2019); Xie et al. (2022).

Statistical hotspot detection. Statistical approaches define hotspots via explicit significance tests, evaluating whether observed event density exceeds an expected baseline Tang et al. (2017; 2019); Xie et al. (2020). Many methods score candidate regions using density-based or likelihood-based functions Tang et al. (2019; 2017). Object-centered approaches, for example, scan regions across multiple radii to identify hotspots Liu et al. (2023). Some methods validate statistical significance through randomization, such as Monte Carlo simulation Xie et al. (2020). While they provide strong statistical evidence, such methods can be expensive due to exhaustive region/radius enumeration and repeated validation, which limits scalability on large networks Liu et al. (2023). This motivates learning a surrogate that approximates the outputs of a statistical procedure rather than executing that procedure at inference time.

Learning-based spatial modeling. Graph learning combines node features and connectivity to model dependencies in network data Ying et al. (2018); Hu et al. (2020); Zhang & Cheng (2020). Applications include crime occurrence prediction Wang et al. (2025), police-call prediction Sui et al. (2024), traffic flow forecasting Kong et al. (2024), and urban mobility optimization Khetarpaul & Sharan (2026). Related spatial modeling approaches include spatial regression Qu et al. (2024) and kernel-based intensity estimation Romano & Jiang (2017). These applications provide context for spatial prediction, but predictive performance alone does not establish agreement with a statistical hotspot detector. Our task instead concerns reproducing detector outputs for an observed event set, including hotspot membership and radius.

Gap in existing approaches. Classical statistical hotspot detection provides significance-driven identification, but its computational cost limits scalability on large spatial networks. Existing learning-based methods primarily predict event intensity, risk, or hotspot scores, rather than reproduce the outputs of a statistical detector. This leaves a gap in efficiently detecting both the membership and spatial radius of statistically defined hotspots. Inspired by neural surrogate modeling Hua et al. (2023); McCabe et al. (2024), we bridge this gap using a graph-based surrogate for a network local K-function–based detector Liu et al. (2023). The model jointly predicts hotspot membership and radius, replacing repeated region and radius searches with learned inference. The statistical detector provides the ground-truth hotspot labels and radii, and agreement on the same evaluated events measures how faithfully the surrogate reproduces them.

3 Problem Formulation

A hotspot is an event-centered network neighborhood with a statistically significant concentration of events, as determined by the statistical procedure. We formulate the problem of learning hotspot detection for spatial events located on a spatial road network as follows.

Input: (1) $G_O = (V_O, E_O)$ denotes a spatial road network, where V_O represents intersections and E_O represents road segments connecting them. Distances between network locations are measured along shortest paths on the road network. (2) $\mathcal{E}_t = \{e_{t1}, \ldots, e_{tn}\}$ denotes a set of training events,

each associated with a location along a road segment. (3) $\mathcal{E}_b = \{E_{b1}, E_{b2}, \ldots, E_{bm}\}$ denotes batches of new events for prediction. Each batch E_{bj} is located on its corresponding road network $G_{O,j}$.

Output: (1) A surrogate model $f_{\theta^\star}$ trained using events in $\mathcal{E}_t$ to jointly predict hotspot membership and the associated radius. For an event set $\mathcal{E}$ on road network G_O,

$$f_{\theta^\star}(G_O, \mathcal{E}) = \{(\hat{y}_i, \hat{r}_i)\}_{i=1}^{|\mathcal{E}|}.$$

(2) Predictions for each new event batch E_{bj} on road network $G_{O,j}$:

$$f_{\theta^\star}(G_{O,j}, E_{bj}) = \{(\hat{y}_{ji}, \hat{r}_{ji})\}_{i=1}^{|E_{bj}|}, \qquad j = 1, \ldots, m.$$

Objective: We learn the model parameters by minimizing a joint objective for hotspot membership and radius estimation:

$$\theta^\star = \arg\min_{\theta} \left[\mathcal{L}_{\text{cls}}(\theta) + \lambda \mathcal{L}_{\text{reg}}(\theta)\right].$$

During prediction, the learned model is applied to new spatial data without requiring statistically generated labels. $\hat{y}_i \in \{0, 1\}$ denotes predicted hotspot membership for event e_i, and $\hat{r}_i \in \mathbb{R}^+$ denotes its associated radius estimate. The model learns from statistically derived labels and radii to approximate the detector's outputs while enabling efficient inference on large spatial networks.

For training data, a network local K-function-based procedure assigns each event e_i a binary hotspot membership label $y_i \in \{0, 1\}$. For a hotspot event ($y_i = 1$), the procedure also returns an associated radius $r_i \in \mathbb{R}^+$, measured along the road network. This radius describes the extent of the significant neighborhood centered at that event. For non-hotspot events ($y_i = 0$), the radius target is undefined and excluded from the regression objective.

The classification term measures disagreement in hotspot membership across training events. The regression term measures radius prediction error only for statistically identified hotspot events. The coefficient λ balances the two tasks, with detailed loss definitions provided in the Learning Objective subsection.

4 METHODOLOGY

4.1 STATISTICAL SUPERVISION FROM THE NETWORK LOCAL K-FUNCTION

We generate ground-truth hotspot labels and radii using the network local K-function–based detector Liu et al. (2023). For an event vertex ϑ and network distance t, the local K-function is defined as

$$K(t \mid \vartheta) = \frac{N(t \mid \vartheta)}{\rho},$$

where $N(t \mid \vartheta)$ denotes the number of event vertices reachable within network distance t from ϑ, and $\rho = \frac{|V_e|-1}{L(G)}$ is the overall event density in the spatial network. Here, $L(G)$ is the total network length.

Under the complete spatial randomness (CSR) hypothesis, the expected value and variance of the local K-function are determined by the length of the isodistance sub-network around ϑ. For a radius t, an event vertex ϑ is identified as a hotspot when its observed local K-function exceeds the α-quantile of the corresponding null distribution, i.e.,

$$K(t \mid \vartheta) > K_\alpha(t \mid \vartheta),$$

where α is the minimum statistical significance threshold.

To generate robust statistically grounded training data for learning, we apply the statistical procedure with each event e_i serving as a query center. The binary label $y_i \in \{0, 1\}$ indicates whether the procedure detects a statistically significant neighborhood centered at e_i. For hotspot events, r_i is the network-distance radius returned for the neighborhood centered at that same event. Following the unknown-radius setting in Liu et al. (2023), r_i corresponds to the first local maximum of statistical significance above the minimum threshold. This provides an event-specific target for learning the spatial extent of each detected hotspot.

Table 1: Input features for event nodes V_E and intersection nodes V_O.

Node	Idx	Feature	Description
V_E	1	Segment length	Length of the road segment containing the event
V_E	2	Segment event density	Number of events per unit road length
V_E	3	Distance to endpoint 1	Distance from the event to one segment endpoint
V_E	4	Distance to endpoint 2	Distance from the event to the other segment endpoint
V_O	1	Avg. intersection distance	Average distance to neighboring intersections
V_O	2	Avg. event distance	Average distance to neighboring events
V_O	3	Node degree	Number of adjacent intersections
V_O	4	Neighboring event count	Number of neighboring event nodes

4.2 Graph Construction

We construct a heterogeneous graph $G = (V, E)$ with two node types: intersection nodes V_O and event nodes V_E. Each intersection in the spatial network is represented as an O-node, and each event mapped to a road segment is represented as an E-node. We include three edge types: $O-O$ edges, which preserve the original network connectivity, and $E-O$ and $O-E$ edges, which connect each event node to the two endpoint intersections of the segment on which it occurs. Formally,

$$V = V_O \cup V_E, \quad E = E_{OO} \cup E_{EO} \cup E_{OE}.$$

Our construction preserves the topology of the underlying spatial network while retaining individual events as prediction targets. Adding event–event edges allows neighboring events to exchange information directly, but also increases the number of edges stored and processed. Connecting all pairs of events within a neighborhood makes the number of these edges grow quadratically with the number of events in that neighborhood, which significantly limits support for large-scale datasets. We instead model event interactions through shared intersection nodes and local density features. Each event has only two intersection neighbors, so the number of event-related edges grows linearly with the number of events. This provides a sparse representation for learning event-level hotspot membership and radius.

4.3 Node Features

Table 1 defines feature vectors for both event nodes V_E and intersection nodes V_O to capture spatial geometry and local event density as follows.

Event node features. Event features capture road scale, within-segment position, and local event density, while intersection features summarize local topology and nearby event concentration. Distance-based features use $\log(1 + x)$; density/count features use percentile ranks within each graph to improve cross-city comparability.

Each event and intersection node has four input features, each transformed appropriately for its scale. These transformations improve optimization stability and enable cross-city generalization without target-city labels or retraining.

4.4 Graph Neural Network Model

We employ a heterogeneous graph neural network (GNN) to learn representations over the constructed graph $G = (V_O \cup V_E, E)$. The model operates on both intersection nodes and event nodes, propagating information through the network via message passing.

Message Passing. Let $h_v^{(k)} \in \mathbb{R}^d$ denote the representation of node v at layer k, with $h_v^{(0)}$ initialized from the input features. Since the graph is heterogeneous, messages are computed separately for each edge type. For a node v, we first aggregate relation-specific messages as

$$m_v^{(k,r)} = \frac{1}{|\mathcal{N}_r(v)|} \sum_{u \in \mathcal{N}_r(v)} h_u^{(k)},$$

where $r \in \{OO, EO, OE\}$ denotes an edge type and $\mathcal{N}_r(v)$ is the set of in-neighbors of v under relation r.

The representation of node v is then updated by summing, over all relation types pointing to v, a relation-specific transformation of the aggregated message and of its previous representation:

$$h_v^{(k+1)} = \phi\left(\sum_{r\in\mathcal{R}(v)}\left(W_r^{(k)} m_v^{(k,r)} + U_r^{(k)} h_v^{(k)} + b_r^{(k)}\right)\right),$$

where $\mathcal{R}(v)$ is the set of relation types pointing to v, $W_r^{(k)}$, $U_r^{(k)}$ and $b_r^{(k)}$ are learnable relation-specific parameters, and ϕ is the ReLU activation, omitted after the last layer. The four-layer backbone thus has 76,032 parameters and the two heads add 4,290 (Table 5).

Shared Backbone. We use a multi-layer GraphSAGE-style heterogeneous backbone, in which the outputs of relation-specific GraphSAGE convolutions are summed at each node. The resulting node embeddings are shared across the downstream classification and regression heads.

Multi-task Prediction Heads. For event nodes $v \in V_E$, final embeddings h_v feed linear classification and multilayer perceptron (MLP) regression heads:

$$z_v = \mathrm{Linear}_{\mathrm{cls}}(h_v), \qquad \hat{\tilde{r}}_v = \mathrm{MLP}_{\mathrm{reg}}(h_v),$$

where z_v is the hotspot logit and $\hat{\tilde{r}}_v$ is the predicted normalized log-radius (Section 4.5).

This architecture enables joint modeling of hotspot membership and hotspot radius from shared node representations. By propagating information across both event and intersection nodes, the model captures local event density as well as broader network structure, which are both essential for approximating statistically grounded hotspot detection.

4.5 Learning Objective

We train the model using supervision derived from the statistical hotspot detection procedure as described in Section 4.1. This data provides binary hotspot labels for all event nodes and, for hotspot events, associated radii.

Target normalization. The radius values r_i exhibit a wide dynamic range across datasets. To stabilize training, we transform and normalize the radius targets as:

$$\tilde{r}_i = \frac{\log r_i - \mu}{s},$$

where μ and s are the mean and standard deviation of $\log r_i$ computed over ground-truth hotspots in the training data. Predicted radii are mapped back to meters as $\hat{r}_i = \exp(\mu + s\,\hat{\tilde{r}}_i)$.

Classification loss. To address class imbalance in hotspot labels, we use a class-weighted binary cross-entropy loss:

$$\mathcal{L}_{\mathrm{cls}} = -\frac{1}{N}\sum_{i=1}^{N} w_{y_i}\left[y_i \log p_i + (1-y_i)\log(1-p_i)\right], \qquad p_i = \mathrm{sigmoid}(z_i),$$

where p_i is the predicted hotspot probability and w_{y_i} is a class-dependent weight: $w_1 = w_+$ for positive (hotspot) samples and $w_0 = w_-$ for negative. The class weights control the relative contributions of hotspot and non-hotspot errors and are held fixed across models.

Regression loss. We train the radius prediction using Smooth L1 over positive samples only:

$$\mathcal{L}_{\mathrm{reg}} = \frac{1}{|\{i : y_i = 1\}|}\sum_{i:y_i=1} \mathrm{SmoothL1}(\hat{\tilde{r}}_i, \tilde{r}_i),$$

where

$$\mathrm{SmoothL1}(x,y) = \begin{cases} \frac{1}{2}(x-y)^2, & \text{if } |x-y| < 1, \\ |x-y| - \frac{1}{2}, & \text{otherwise.} \end{cases}$$

The regression loss is restricted to hotspot events, as non-hotspot events do not have a meaningful radius.

Multi-task objective. The final training objective combines both tasks:

$$\mathcal{L} = \mathcal{L}_{\text{cls}} + \lambda \mathcal{L}_{\text{reg}},$$

where λ controls the trade-off between classification and regression.

Inference. At inference time, the predicted label is $\hat{y}_i = \mathbf{1}[p_i \geq \eta]$, where the threshold η is selected on the validation set, and the predicted radius is $\hat{r}_i$. Our primary output is event-level hotspot membership rather than an explicit global partition of the network into hotspot regions. In other words, the classifier determines whether each event node is identified as a hotspot, while the regression head estimates the characteristic radius associated with that hotspot.

Threshold Calibration. We provide an optional calibration step that combines GNN predictions with selective statistical verification to accommodate different application requirements. For example, police departments may prioritize detecting as many crime hotspots as possible, favoring recall over precision. Lowering the prediction threshold increases the recall, but also introduces more false positives. Statistical verification filters out the additional that do not qualify as hotspots.

We select an upper and lower values for the prediction threshold, depending on application requirements. The lower value corresponds to a target recall and the upper value corresponds to a target precision. This pair of values is fixed across all test graphs without using target-city labels or retraining the model. Events below the lower threshold are discarded, and those at or above the upper threshold are accepted directly. Events between the values are evaluated by statistical method, i.e., AH-IBT, and retained only if they pass its significance test. The hotspot radius prediction remain intact.

5 Experiments

We evaluate the proposed framework along three axes: **(Q1)** Does GNN inference scale to datasets with millions of events? **(Q2)** How accurately does the learned surrogate approximate the statistical method? **(Q3)** Can a model trained on one city generalize zero-shot to unseen spatial networks?

5.1 Experimental Setup

Datasets and evaluation protocol. We evaluate on four spatial-network datasets constructed from public crime event records and service calls City of Chicago; City of Los Angeles. The datasets span substantially different event scales, from 214K events in Seattle to 8M in the main Los Angeles test graph, and road-network sizes from 11K to 36K intersections. Ground-truth hotspot labels and radii are produced by the exact network local K-function detector with adaptive radius search Liu et al. (2023) at a minimum significance threshold $\alpha = 0.94$; AH-IBT Liu et al. (2023), its faster approximation, serves as the runtime baseline and the verifier in calibration. Table 2 summarizes the main test graphs.

Table 2: Main test-graph statistics.

City	$\lvert V_O \rvert$	$\lvert E_{OO} \rvert$	$\lvert V_E \rvert$	Hotspot %	Role
Los Angeles	35,975	56,633	8,000,000	59.9	In-city test
Chicago	23,292	40,787	7,679,723	63.9	Zero-shot test
Detroit	16,259	28,538	5,171,281	73.3	Zero-shot test
Seattle	11,386	19,348	213,645	74.2	Zero-shot test

Los Angeles provides 500K training events, 500K validation events, and an 8M-event main test graph drawn from a separate 16M-event pool; Chicago, Detroit, and Seattle are reserved for zero-shot transfer. For scalability, we evaluate LA test graphs with $|V_E| \in \{20\text{K}, 100\text{K}, 250\text{K}, 500\text{K}, 1\text{M}, 8\text{M}\}$. We also evaluate five test subsets per city for Chicago and Detroit, and three for Seattle, in addition to their full graphs. The 22 test graphs and their results are listed in Table 6; subsets and full graphs are not independent replicates.

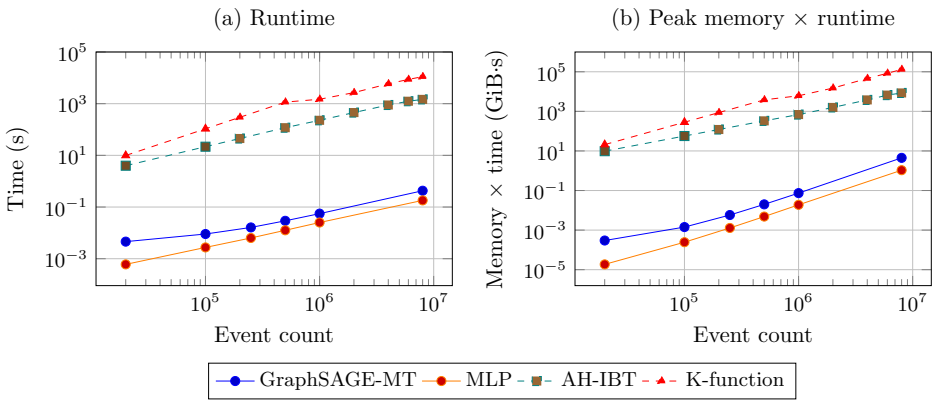


Figure 1: Inference scalability on the LA road network **(a)** Mean forward time. **(b)** Peak memory × runtime.

Models. Our primary model is **GraphSAGE-MT**, a heterogeneous GraphSAGE Hamilton et al. (2017) backbone with multi-task classification and regression heads. We compare against a graph-free **MLP** using the same event-node features only, trained with the same train/validation files and loss weights. Model configurations are reported in Table 5. The GNN is trained to approximate these statistical labels; detection quality metrics therefore measure approximation fidelity to them, not improvement over the statistical method.

Metrics. For classification, we report F1 score, precision, recall, and average precision (AP, computed without interpolation). Unless otherwise stated, GraphSAGE-MT uses a threshold of 0.399, selected to maximize F1 on the LA validation set. The MLP retains its original evaluation setting (Appendix A.1). For radius estimation, we report mean absolute error (MAE, in meters) and mean absolute percentage error (MAPE), computed over events labeled as hotspots by the statistical method only. For scalability, we report CUDA-synchronized forward time, averaged over three runs after one warm-up, together with peak memory × runtime (GiB·s) to compare resource usage. Timing excludes data loading, graph construction, host-to-device transfer, and metric computation.

Implementation. Models are implemented in PyTorch, with PyTorch Geometric for GraphSAGE-MT, and are trained with Adam for up to 500 epochs with early stopping, using a patience of 50 epochs and monitoring validation loss. The multi-task objective combines class-weighted binary cross-entropy for classification with shared class weights and Smooth L1 loss for radius regression, with balancing coefficient $\lambda = 1.0$ (Section 4.5). Inference measurements use one NVIDIA RTX 3090 (24 GiB VRAM), 32 GiB host memory, and four CPU cores. We evaluate one checkpoint per model selected by minimum validation loss; training and inference use PyTorch 2.5.1 and PyTorch Geometric 2.7.0.

5.2 Experimental Results

Inference Scalability. Figure 1 reports runtime and peak memory × runtime as the number of events grows from 20K to 8M on the fixed LA road network. On 8M events, GraphSAGE-MT takes **0.427 s** of forward time, about 3,428× and 25,979× faster than AH-IBT and the K-function. Peak memory is of the same order across methods (5.81 GiB for AH-IBT and 11.84 GiB for the K-function at 8M events), but the statistical methods hold that memory for orders of magnitude longer. As a result, GraphSAGE-MT's peak memory × runtime is about 1,905× and 29,390× lower than that of AH-IBT and the K-function. Its forward time grows approximately linearly at the larger tested scales.

Detection Quality. Table 3 reports approximation quality on the LA test set (8M events). GraphSAGE-MT achieves F1 = 0.897, AP = 0.979, and radius MAE = 22.1 m. The MLP baseline achieves F1 = 0.806 and radius MAE = 75.3 m under the same data and loss settings.

Cross-City Generalization. An advantage of the learned surrogate over statistical methods is that it can be applied to new spatial networks at inference time without re-running the statistical detector. Table 4 evaluates GraphSAGE-MT, trained solely on Los Angeles, on LA and three unseen cities. At

Table 3: Detection quality against the statistical labels on LA 8M.

Model	F1	Prec.	Recall	AP	MAE (m)	MAPE
MLP (no graph)	0.806	0.944	0.703	0.942	75.3	26.9%
GraphSAGE-MT	0.897	0.959	0.842	0.979	22.1	8.5%

Table 4: Cross-city performance of GraphSAGE-MT trained on LA.

City	F1	Prec.	Recall	MAE (m)	MAPE
Los Angeles	0.897	0.959	0.842	22.1	8.5%
Chicago	0.889	0.978	0.815	13.1	6.6%
Detroit	0.884	0.965	0.815	24.6	13.3%
Seattle	0.939	0.968	0.912	21.0	11.1%

the default threshold, F1 remains at least 0.884 across all cities, reaching 0.939 on Seattle. Radius MAE ranges from 13.1 m on Chicago to 24.6 m on Detroit, corresponding to MAPE of 6.6–13.3%.

The GNN processes each main test graph in under 0.43 s of GPU forward time. On the three unseen cities, precision ranges from 0.965 to 0.978 and recall from 0.815 to 0.912.

GraphSAGE uses the same neighborhood aggregation across cities, with consistent node types, relations, and feature definitions. Combined with labels generated by the same statistical procedure, this enables transfer across spatial networks.

Feature ablation. We retrain GraphSAGE-MT without intersection or event features (Appendix A.3). Removing intersection features mainly hurts cross-city transfer (Detroit recall 0.815→0.639, versus 0.842→0.827 on LA), as they encode neighborhood event density as within-city percentiles, a signal comparable across cities. Removing event features raises LA radius MAE from 22.1 to 54.3 m and lowers Seattle specificity from 0.914 to 0.650: events on the same segment then receive identical inputs, so non-hotspot events on dense segments become false positives and radii cannot vary with event position.

Calibration improved recall and precision at the cost of additional statistical computation. We report two configurations selected on the LA validation set and applied unchanged to all test graphs. The first configuration was selected to meet validation recall of at least 98% and precision of at least 99%, yielding thresholds of $(0.10, 0.95)$. On LA 8M, this configuration increased recall from 84.24% to 95.73% and precision from 95.85% to 99.87% compared to previous GNN performance. It verified 2.65 million events in only 83.56 s, compared with 452.95 s for full AH-IBT on 2 million events. AH-IBT only determines hotspot membership, avoiding the overhead of radius search.

Maximizing validation F1 selected thresholds of $(0.01, 1.00)$. This configuration achieved 99.84% recall and approximately 100% precision on LA 8M, with 85.38% of events verified in 190.85 s. These results show that closer agreement with the statistical detector can be obtained through a larger verification workload. Full results are reported in Tables 8 and 9 in Appendix A.4.

6 Conclusion

We introduce a scalable GNN surrogate for event-level statistically grounded hotspot detection on spatial networks. Using supervision from a local K-function method, our model jointly approximates hotspot membership and radius from road-network and event features. On the LA test set with 8 million events, GraphSAGE-MT approximates the statistical labels, with F1 $= 0.897$, AP $= 0.979$, and radius MAE $= 22.1$ meters, with 0.427 seconds of GPU forward time on an RTX 3090. The model also transfers zero-shot to unseen cities, achieving F1 scores from 0.884 to 0.939. The surrogate should be interpreted as an approximation of the statistical detector rather than a replacement for formal statistical inference; future work should study probability calibration, uncertainty-aware verification, and stricter spatial or temporal holdout evaluation.

AI use statement

In this work, we used generative AI tools to provide feedback on the research methodology and experiments, to implement parts of the experimental pipeline (evaluation, calibration, and timing

scripts), and to assist in interpreting results. We have not used generative AI tools to generate synthetic data, and formulating mathematical claims or writing proofs is not applicable to this work. Additionally, we used generative AI tools to brainstorm research directions, search for and summarize related literature, draft parts of the paper, edit the text for readability, create figures, and check citations and reported numbers against saved experimental records. We have reviewed all AI-assisted work: AI-generated code and its outputs were checked by the authors against saved experimental records, cited references were verified against their original sources, and AI-drafted text was revised by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility statement

Section 5.1 describes the datasets, splits, evaluation protocol, software and hardware, and Appendix A.1 lists all model configurations. Results on every test graph, the feature ablation and the calibration settings are given in Appendices A.2, A.3 and A.4. The anonymized supplementary code implements every step with the exact commands and split parameters: seeded data splitting, statistical label generation ($\alpha = 0.94$), graph construction, training, evaluation (threshold selection and timing), calibration and the statistical baselines. The underlying event records are public (City of Chicago; City of Los Angeles).

References

City of Chicago. Chicago data portal. `https://data.cityofchicago.org/`.

City of Los Angeles. City of los angeles open data catalog. `https://catalog.data.gov/organization/city-of-los-angeles`.

Min Deng, Xuexi Yang, Yan Shi, Jianya Gong, Yang Liu, and Huimin Liu. A density-based approach for detecting network-constrained clusters in spatial point events. *International Journal of Geographical Information Science*, 33(3):466–488, 2019.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pp. 226–231, 1996.

William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. NIPS'17, pp. 1025–1035, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Alexander Hohl, Eric M Delmelle, Michael R Desjardins, and Yu Lan. Daily surveillance of covid-19 using the prospective space-time scan statistic in the united states. *Spatial and spatio-temporal epidemiology*, 34:100354, 2020.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 22118–22133. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper_files/paper/2020/file/fb60d411a5c5b72b2e7d3527cfc84fd0-Paper.pdf`.

Chuanbo Hua, Federico Berto, Michael Poli, Stefano Massaroli, and Jinkyoo Park. Learning efficient surrogate dynamic models with graph spline networks. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL `https://openreview.net/forum?id=MGPST5I9DO`.

Shoaib Khalid, Fariha Shoaib, Tianlu Qian, Yikang Rui, Arezu Imran Bari, Muhammad Sajjad, Muhammad Shakeel, and Jiechen Wang. Network constrained spatio-temporal hotspot mapping of crimes in faisalabad. *Applied Spatial Analysis and Policy*, 11(3):599–622, 2018.

Sonia Khetarpaul and PY Sharan. Traffic-aware optimal taxi placement using graph neural network-based reinforcement learning. *arXiv preprint arXiv:2601.00607*, 2026.

Weiyang Kong, Ziyu Guo, and Yubao Liu. Spatio-temporal pivotal graph neural networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 8627–8635, 2024.

Yongyi Liu, Yunfan Kang, Ahmed Mahmood, and Amr Magdy. Scalable evaluation of local k-function for radius-accurate hotspot detection in spatial networks. In *Proceedings of the 31st ACM International Conference on Advances in Geographic Information Systems*, pp. 1–12, 2023.

Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2): 129–137, 1982.

Michael McCabe, Bruno Régaldo-Saint Blancard, Liam Holden Parker, Ruben Ohana, Miles Cranmer, Alberto Bietti, Michael Eickenberg, Siavash Golkar, Geraud Krawezik, Francois Lanusse, Mariel Pettee, Tiberiu Tesileanu, Kyunghyun Cho, and Shirley Ho. Multiple physics pretraining for spatiotemporal surrogate models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=DKSI3bULiZ.

Xinyu Qu, Xiongwu Xiao, Xinyan Zhu, Zhenfeng Shao, Mi Wang, Huayi Wu, Hongkai Zhao, Jianya Gong, and Deren Li. St-gwlr: combining geographically weighted logistic regression and spatiotemporal hotspot trend analysis to explore the effect of built environment on traffic crash. *Geo-Spatial Information Science*, 27(4):1017–1034, 2024.

Benjamin Romano and Zhe Jiang. Visualizing traffic accident hotspots based on spatial-temporal network kernel density estimation. In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 1–4, 2017.

Yan Shi, Yuanfang Chen, Min Deng, Liang Xu, and Jiaqin Xia. Discovering source areas of disease outbreaks based on ring-shaped hotspot detection in road network space. *International Journal of Geographical Information Science*, 36(7):1343–1363, 2022.

Jinguang Sui, Peng Chen, and Haishuo Gu. Deep spatio-temporal graph attention network for street-level 110 call incident prediction. *Applied Sciences*, 14(20):9334, 2024.

Xun Tang, Emre Eftelioglu, and Shashi Shekhar. Detecting isodistance hotspots on spatial networks: A summary of results. In *International Symposium on Spatial and Temporal Databases*, pp. 281–299. Springer, 2017.

Xun Tang, Jayant Gupta, and Shashi Shekhar. Linear hotspot discovery on all simple paths: A summary of results. In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 476–479, 2019.

Shun Wang, Yong Zhang, Xinglin Piao, Xuanqi Lin, Yongli Hu, and Baocai Yin. Mragnn: Refining urban spatio-temporal prediction of crime occurrence with multi-type crime correlation learning. *Expert Systems with Applications*, 265:125940, 2025.

Tianfu Wang, Chang Ren, Yun Luo, and Jing Tian. Ns-dbscan: A density-based clustering algorithm in network space. *ISPRS International Journal of Geo-Information*, 8(5):218, 2019.

Yiqun Xie and Shashi Shekhar. A nondeterministic normalization based scan statistic (nn-scan) towards robust hotspot detection: A summary of results. In *Proceedings of the 2019 SIAM International Conference on Data Mining*, pp. 82–90. SIAM, 2019.

Yiqun Xie, Xun Zhou, and Shashi Shekhar. Discovering interesting subpaths with statistical significance from spatiotemporal datasets. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(1):1–24, 2020.

Yiqun Xie, Shashi Shekhar, and Yan Li. Statistically-robust clustering techniques for mapping spatial hotspots: A survey. *ACM Computing Surveys (CSUR)*, 55(2):1–38, 2022.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 974–983, 2018.

Yang Zhang and Tao Cheng. Graph deep learning model for network-based predictive hotspot mapping of sparse spatio-temporal events. *Computers, Environment and Urban Systems*, 79: 101403, 2020.

A Additional Experimental Details

A.1 Model Configurations

Table 5 summarizes the model configurations used in the experiments. Here, L is the number of layers, d is the hidden dimension, H is the number of attention heads, and FB denotes full-batch training. The MLP retains its best F1 decision threshold of 0.5.

Table 5: Model configurations.

Model	L	d	H	**Dropout**	**Loader**	**Params**
MLP	4	64	—	0.16	FB	17,090
GraphSAGE-MT	4	64	—	0.16	FB	80,322

A.2 Results Across Test Graphs

Table 6 reports results on all 22 test graphs using thresholds of 0.399 for GraphSAGE-MT and 0.5 for MLP. Radius MAE is measured in meters on labeled hotspots.

Table 6: GraphSAGE-MT and MLP performance on all 22 test graphs.

		GraphSAGE-MT			MLP		
City	Events	F1	AP	MAE	F1	AP	MAE
LA	20,000	0.788	0.898	28.0	0.719	0.843	76.1
LA	100,000	0.866	0.956	24.6	0.787	0.892	82.1
LA	250,000	0.896	0.969	23.4	0.806	0.912	79.5
LA	500,000	0.908	0.974	23.1	0.814	0.922	78.9
LA	1,000,000	0.909	0.977	22.7	0.815	0.930	77.3
LA	8,000,000	0.897	0.979	22.1	0.806	0.942	75.3
Chicago	20,000	0.839	0.925	19.0	0.750	0.850	56.1
Chicago	100,000	0.905	0.969	15.4	0.794	0.909	53.9
Chicago	250,000	0.915	0.980	14.0	0.805	0.933	51.5
Chicago	500,000	0.910	0.983	13.4	0.805	0.942	50.5
Chicago	1,000,000	0.905	0.984	13.2	0.802	0.950	50.0
Chicago	7,679,723	0.889	0.986	13.1	0.788	0.958	49.6
Detroit	20,000	0.876	0.956	27.8	0.818	0.948	50.2
Detroit	100,000	0.895	0.972	27.0	0.804	0.961	49.2
Detroit	250,000	0.895	0.975	25.3	0.794	0.965	49.3
Detroit	500,000	0.893	0.976	25.1	0.788	0.967	48.7
Detroit	1,000,000	0.889	0.977	24.8	0.782	0.969	48.4
Detroit	5,171,281	0.884	0.978	24.6	0.768	0.972	47.9
Seattle	10,000	0.895	0.965	25.2	0.746	0.916	59.0
Seattle	20,000	0.917	0.977	23.4	0.753	0.935	58.2
Seattle	50,000	0.936	0.985	22.0	0.754	0.952	57.9
Seattle	213,645	0.939	0.991	21.0	0.742	0.964	55.4

A.3 Feature Ablation

Table 7: Feature ablation on the main test graphs.

City	Model	F1	Prec.	Recall	Spec.	AP	MAE (m)	MAPE
Los Angeles	Full	0.897	0.959	0.842	0.946	0.979	22.1	8.5%
	w/o intersection features	0.881	0.941	0.827	0.923	0.970	38.1	14.3%
	w/o event features	0.893	0.951	0.841	0.935	0.975	54.3	26.2%
Chicago	Full	0.889	0.978	0.815	0.967	0.986	13.1	6.6%
	w/o intersection features	0.868	0.965	0.789	0.949	0.977	27.1	13.8%
	w/o event features	0.898	0.958	0.846	0.934	0.979	37.5	24.2%
Detroit	Full	0.884	0.965	0.815	0.920	0.978	24.6	13.3%
	w/o intersection features	0.777	0.992	0.639	0.986	0.976	24.5	12.8%
	w/o event features	0.903	0.906	0.900	0.743	0.967	56.7	31.7%
Seattle	Full	0.939	0.968	0.912	0.914	0.991	21.0	11.1%
	w/o intersection features	0.854	0.990	0.750	0.979	0.988	33.0	13.8%
	w/o event features	0.932	0.890	0.979	0.650	0.988	47.4	25.0%

A.4 CALIBRATION

Table 8: Calibration results with thresholds $(0.10, 0.95)$.

City	Events	Recall (%)	Precision (%)	F1	Verified (%)	Verification time (s)
LA	20,000	97.93	91.6884	0.9471	42.30	4.68
LA	100,000	98.96	97.6133	0.9828	37.09	8.27
LA	250,000	98.69	99.2388	0.9897	35.13	10.47
LA	500,000	98.38	99.5957	0.9898	34.04	13.58
LA	1,000,000	97.75	99.7104	0.9872	33.30	17.96
LA	8,000,000	95.73	99.8663	0.9775	33.09	83.56
Chicago	20,000	97.98	94.1970	0.9605	36.20	3.02
Chicago	100,000	98.48	98.8173	0.9865	31.72	5.19
Chicago	250,000	97.76	99.6557	0.9870	30.47	6.87
Chicago	500,000	96.87	99.8431	0.9834	30.04	8.96
Chicago	1,000,000	96.00	99.9039	0.9791	29.72	13.06
Chicago	7,679,723	93.76	99.9623	0.9676	29.71	65.89
Detroit	20,000	95.72	98.7876	0.9723	40.00	2.52
Detroit	100,000	96.46	99.5206	0.9797	41.07	4.62
Detroit	250,000	96.01	99.7027	0.9782	39.31	5.97
Detroit	500,000	95.75	99.7541	0.9771	39.35	8.53
Detroit	1,000,000	95.25	99.7886	0.9747	38.65	13.06
Detroit	5,171,281	94.61	99.8665	0.9717	39.14	54.67
Seattle	10,000	98.24	98.3896	0.9831	38.18	2.11
Seattle	20,000	98.45	99.0400	0.9875	35.95	2.84
Seattle	50,000	98.73	99.6395	0.9918	35.46	3.90
Seattle	213,645	98.62	99.9387	0.9928	34.58	7.06

Table 9: Calibration results with thresholds $(0.01, 1.00)$.

City	Events	Recall (%)	Precision (%)	F1	Verified (%)	Verification time (s)
LA	20,000	99.92	100.000000	0.9996	94.84	7.42
LA	100,000	99.93	100.000000	0.9996	89.83	16.34
LA	250,000	99.93	100.000000	0.9996	87.50	22.21
LA	500,000	99.92	100.000000	0.9996	86.29	29.09
LA	1,000,000	99.90	100.000000	0.9995	85.81	41.00
LA	8,000,000	99.84	99.999184	0.9992	85.38	190.85
Chicago	20,000	99.91	100.000000	0.9996	90.56	7.28
Chicago	100,000	99.91	100.000000	0.9995	81.84	10.92
Chicago	250,000	99.90	100.000000	0.9995	80.20	14.47
Chicago	500,000	99.89	100.000000	0.9994	79.84	19.99
Chicago	1,000,000	99.86	100.000000	0.9993	79.46	29.19
Chicago	7,679,723	99.72	99.999857	0.9986	79.26	157.87
Detroit	20,000	99.63	100.000000	0.9982	96.77	4.75
Detroit	100,000	99.76	100.000000	0.9988	97.05	8.90
Detroit	250,000	99.76	100.000000	0.9988	96.87	12.60
Detroit	500,000	99.80	100.000000	0.9990	96.82	18.25
Detroit	1,000,000	99.79	100.000000	0.9989	96.74	28.51
Detroit	5,171,281	99.76	100.000000	0.9988	96.76	115.54
Seattle	10,000	99.88	100.000000	0.9994	96.08	3.66
Seattle	20,000	99.87	100.000000	0.9994	95.05	4.73
Seattle	50,000	99.89	100.000000	0.9995	94.06	6.93
Seattle	213,645	99.87	100.000000	0.9994	93.23	14.16