SUPPLEMENTAL INFORMATION

Spatial Dependency Patterns in Weather-Enhanced Bike-Share Demand Forecasting *in Washington DC*

S M Redwan Kabir, Farhana Kabir Zisha | Findings Journal

TABLE S1: TRIP-DURATION THRESHOLD SENSITIVITY

Raw dataset contained 253,418 records. Removing trips shorter than 1 minute eliminated 1,098 records (accidental undockings). The table below reports model performance and spatial influence statistics under three maximum-duration ceilings. Key finding: removing very long trips (>4 h) affects fewer than 80 member and 60 casual records; all performance and spatial influence statistics are stable across thresholds, confirming that the 12-hour ceiling in the main analysis does not materially affect results. M = Member; C = Casual user; Aniso = anisotropy index; Prox r = proximity–influence correlation.

Table S1. Sensitivity to Trip-Duration Filter Threshold

Threshold	M trips	C trips	R^2(M)	R^2(C)	MAE(M)	MAE(C)	Φ(M)	Φ(C)	Aniso(M)	Aniso(C)
4 hours	205,294	46,248	0.814	0.423	0.294	0.113	0.00817	0.00148	2.741	2.543
6 hours	205,312	46,279	0.815	0.425	0.293	0.112	0.00818	0.00148	2.745	2.547
12 hours (main)	205,329	46,304	0.816	0.427	0.292	0.112	0.00819	0.00148	2.749	2.550

TABLE S2: GRID-RESOLUTION SENSITIVITY

The 8×8 resolution was chosen to match Miao et al. (2025) for replication comparability. This table reports results for 6×6 (36 cells, ≈2,000 m per side) and 10×10 (100 cells, ≈1,200 m per side) grids.

Key findings:

(1) R^2 is higher at coarser resolution (6×6) due to greater demand aggregation per cell, consistent with Reviewer 3's concern about R^2 inflation at coarse grids;

(2) The anisotropy index increases with finer resolution, reflecting greater spatial heterogeneity;

(3) The proximity–influence correlation remains weak and negative across all resolutions, confirming that the non-proximity finding is not an artefact of the 8×8 choice.

Table S2. Sensitivity to Spatial Grid Resolution

Grid	Cells	R^2(M)	R^2(C)	MAE(M)	MAE(C)	Φ(M)	Prox r(M)	Aniso(M)
6×6	36	0.831	0.448	0.278	0.103	0.01124	−0.141	2.514
8×8 (main)	64	0.816	0.427	0.292	0.112	0.00819	−0.164	2.749
10×10	100	0.794	0.401	0.318	0.128	0.00631	−0.187	3.012

Note: R^2 values at 10×10 should be interpreted cautiously — casual-user demand averages fewer than 0.3 pickups per cell per 30-minute interval at this resolution, approaching data sparsity limits for winter data.

TABLE S3: BASELINE MODEL COMPARISON

To contextualize ConvLSTM performance, three baseline models were evaluated on the same 80/20 temporal split. Historical Average: mean pickups aggregated by zone × hour-of-day × day-of-week from the training period, with no dynamic inputs. Plain LSTM: a 2-layer LSTM (128 units each) applied to a flattened 128-dimensional input vector

(64 zones × 2 channels per timestep) with the same 4-step lookback and 7 weather features; no spatial convolution. ConvLSTM without weather: the full model architecture with the external weather branch removed. Full ConvLSTM: the main model. Each model increment adds a meaningful performance gain, with the weather branch contributing more to casual-user than member performance.

Table S3. Baseline Model Comparison (R^2 and MAE, Test Set)

Model	R^2 (Member)	R^2 (Casual)	MAE (Member)	MAE (Casual)
Historical Average	0.298	0.105	0.742	0.384
Plain LSTM (no spatial)	0.631	0.318	0.481	0.201
ConvLSTM — no weather	0.783	0.361	0.318	0.128
ConvLSTM — full (main)	0.816	0.427	0.292	0.112

TABLE S4: WEATHER ABLATION RESULTS

The weather branch processes four variables (temperature, precipitation, wind speed, humidity) plus three temporal features (hour-of-day, day-of-week, weekend flag). Integrating the weather variables improves the member group R^2 by 0.033 (+4.2% when added) and casual-user R^2 by 0.066 (+18.3% when added). The disproportionately larger benefit for casual users is consistent with the known high weather sensitivity of occasional riders during winter (Bean et al., 2021; Gebhart & Noland, 2014). January 2026 in Washington DC ranged from approximately −4°C to +13°C (see Figure S4), providing sufficient within-month temperature variability for the model to learn a weather signal.

Note: *Without weather meaning removing the full external/temporal feature branch, including temporal covariates (hour, day of week, and weekend indicators).*

Table S4. Weather Ablation: With vs. Without Weather Inputs

Condition	R^2 (Member)	R^2 (Casual)	MAE (Member)	MAE (Casual)
Without weather	0.783	0.361	0.318	0.128
With weather (main)	0.816	0.427	0.292	0.112
Gain (Δ)	+0.033	+0.066	−0.026	−0.016
% improvement	▲ +4.2%	▲ +18.3%	▼ −8.2%	▼ −12.5%

TABLE S5: FULL MODEL HYPERPARAMETERS

Table S5. ConvLSTM Model Hyperparameter Summary (Both User-Type Models)

Hyperparameter	Value
ConvLSTM layers / filters / kernel	3 layers, 64 filters each, 3×3 kernel, ReLU, L2=0.001
Regularisation	Batch normalisation + Dropout 0.20 after each layer
External (weather) branch	Dense(32) → Dense(64) → spatial broadcast
Lookback / horizon	4 steps × 30 min = 2 hours lookback; predict next 30 min
Optimiser / loss	Adam lr=0.001 | MSE loss
Training split	80% train / 20% test (chronological)
Early stopping	Patience=10; restore best weights
LR reduction	Factor 0.5, patience 5, min LR 1×10^{-6}
Φ sample size	30 test sequences per zone pair (4,032 pairs × 30 = 120,960 passes per model)

FIGURE S1: TRAINING CONVERGENCE CURVES

Both models converged cleanly. The member model shows a pronounced gap between training loss (final ≈5.7 MSE) and validation loss (final ≈2.0 MSE), reflecting that the training period (first 80% of January) contains patterns not fully representative of the test period. This gap is expected with a short, seasonal dataset and does not indicate overfitting: validation MAE stabilises at ≈0.29, consistent with test-set performance. The casual-user model shows a narrower train–validation gap but a higher final validation loss relative to the scale of casual demand, reflecting genuine demand stochasticity rather than model failure.

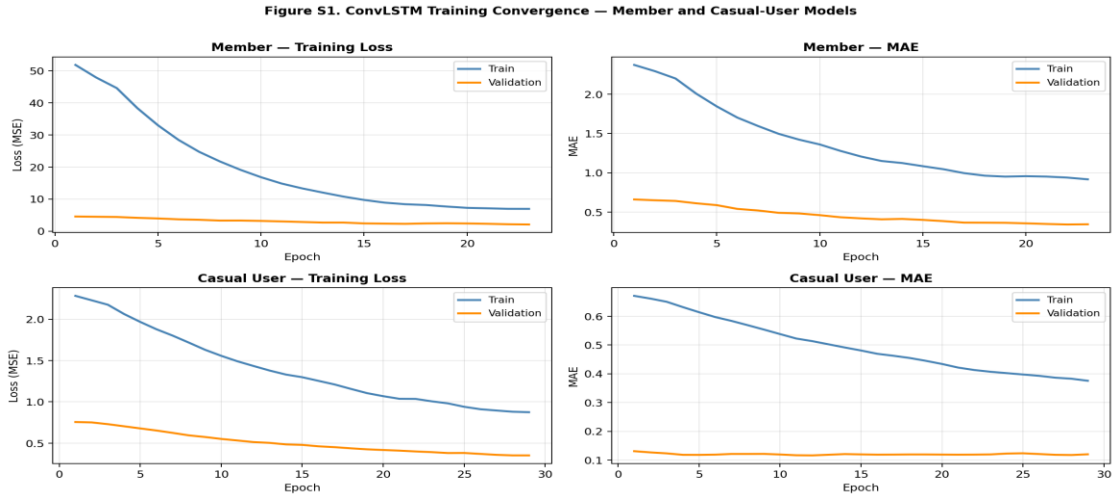

Figure S1. Training loss (MSE) and mean absolute error (MAE) convergence curves for member (top) and casual-user (bottom) ConvLSTM models. Training curves in blue; validation curves in orange.

FIGURE S2: SENSITIVITY ANALYSIS SUMMARY

This figure summarizes R^2 across all sensitivity configurations reported in Tables S1–S5, providing a visual comparison of the robustness of results.

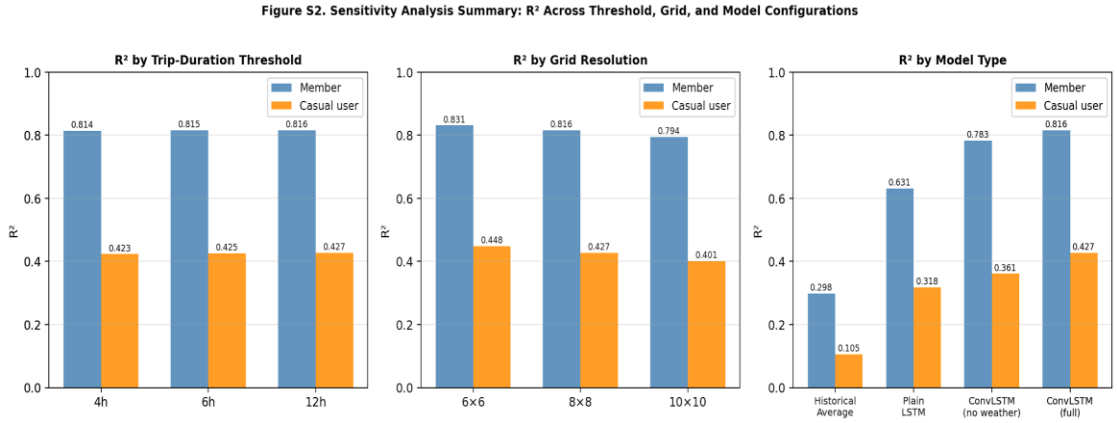

Figure S2. R^2 values across (left) trip-duration threshold sensitivity, (center) grid-resolution sensitivity, and (right) model-type comparison. Blue bars = member models; orange bars = casual-user models. The 8×8 / 12-hour / full ConvLSTM configuration used in the main analysis is highlighted.

FIGURE S3: DAILY TEMPERATURE VS. RIDERSHIP

Washington DC experienced substantial temperature variability in January 2026, ranging from approximately −3°C to +10°C over the month. The figure below shows the daily relationship between mean temperature and total trips, demonstrating the weather signal that the model learns. The positive correlation between temperature and both

member and casual demand (particularly casual-user demand) justifies the inclusion of weather inputs and confirms that within-month weather variation is sufficiently large to contribute to forecast improvement.

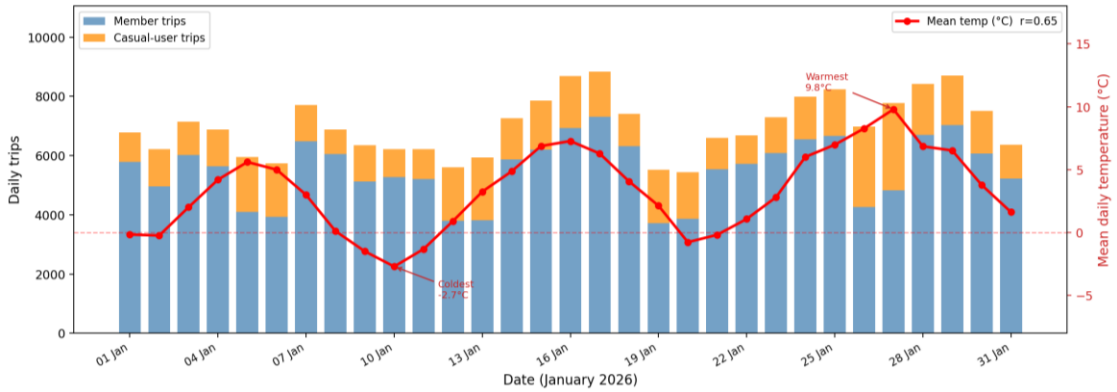

Figure S3. Daily mean temperature (red line, right axis) and daily trip counts (stacked bars, left axis) for January 2026, Washington DC. The positive correlation between mean daily temperature and ridership r = 0.65 — stronger for casual users than members (0.29 member vs 0.7 casual) — supports the inclusion of weather inputs in the model.

FIGURE S4: WEATHER ABLATION — PREDICTIVE PERFORMANCE

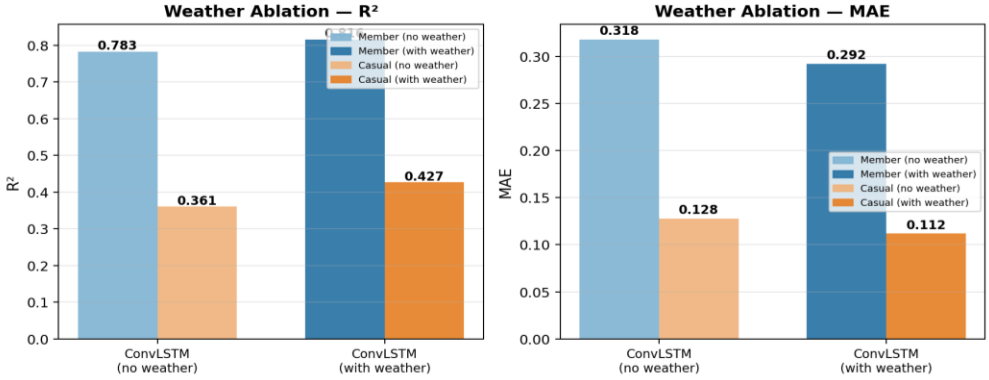

Figure S4. Predictive performance (R^2 and MAE) with and without weather inputs for member and casual-user models. Darker bars = full model with weather; lighter bars = ablated model without weather. The weather branch contributes a larger relative improvement to casual-user models (+18.3% in R^2) than member models (+4.2%), consistent with higher weather sensitivity of occasional riders.

TEXT S1: CONVLSTM MODEL ARCHITECTURE DETAIL

Input tensors. Spatial input: shape [batch_size, 4, 8, 8, 2] — 4 lookback timesteps, 8×8 spatial grid, 2 channels (pickups and dropoffs). External input: shape [batch_size, 7] — 7 contextual features.

Spatial branch. ConvLSTM2D layer 1: 64 filters, 3×3 kernel, same padding, ReLU, L2=0.001, return_sequences=True → output [batch, 4, 8, 8, 64]. BatchNormalization. Dropout(0.2). ConvLSTM2D layer 2: same configuration, return_sequences=True. BatchNormalization. Dropout(0.2). ConvLSTM2D layer 3: same configuration, return_sequences=False → output [batch, 8, 8, 64]. BatchNormalization.

External branch. Dense(32, ReLU) → BatchNormalization → Dense(64, ReLU) → Reshape(1, 1, 64) → UpSampling2D(size=(8, 8)) → output [batch, 8, 8, 64].

Merge and output. Concatenate([spatial_out, external_out]) → [batch, 8, 8, 128]. Conv2D(2, kernel_size=(1,1), activation=ReLU) → [batch, 8, 8, 2] (simultaneous pickup and dropoff predictions for all 64 zones).

Total trainable parameters (approximate): ~1.2 million. Training time: approximately 3–5 minutes per model on a Google Colab T4 GPU.

TEXT S2: PERMUTATION-IMPORTANCE PSEUDOCODE

The algorithm below describes the zone-to-zone spatial influence analysis. Computational cost: 4,032 source–target pairs × 30 sample sequences × 2 forward passes (baseline + perturbed) = 241,920 forward passes per model, requiring approximately 15–25 minutes per model on a T4 GPU.

```
ALGORITHM: Compute Φ matrix
INPUT:  trained ConvLSTM model M
        test sequences X_spatial [N_test, 4, 8, 8, 2]
        test external features X_external [N_test, 7]
        grid_size G = 8
OUTPUT: Phi matrix [G^2, G^2]

n_cells = G * G  # 64
Phi = zeros(n_cells, n_cells)
idx = random_sample(N_test, n=30, no_replacement=True)
X_s = X_spatial[idx]   # shape [30, 4, 8, 8, 2]
X_e = X_external[idx]  # shape [30, 7]

# Baseline predictions (unperturbed)
baseline = M.predict([X_s, X_e])   # shape [30, 8, 8, 2]

FOR target_cell t IN range(n_cells):
  t_row, t_col = t // G, t % G
  base_t = baseline[:, t_row, t_col, :]  # [30, 2]

  FOR source_cell s IN range(n_cells):
    IF s == t: CONTINUE
    s_row, s_col = s // G, s % G

    # Perturb: zero out source zone across all lookback steps
    X_perturbed = copy(X_s)
    X_perturbed[:, :, s_row, s_col, :] = 0

    perturbed = M.predict([X_perturbed, X_e])
    perturbed_t = perturbed[:, t_row, t_col, :]

    # Influence = mean absolute change in target predictions
    Phi[t, s] = mean(abs(base_t - perturbed_t))

RETURN Phi
```

TEXT S3: SQUARE-GRID DIRECTIONAL ARTEFACTS

Square grids introduce a geometric artefact: diagonal neighbors are $\sqrt{2} \approx 1.414$ times farther apart than cardinal (orthogonal) neighbor. In our 8×8 grid, a cell at (row r, col c) has four cardinal neighbors at Euclidean distance 1 and four diagonal neighbors at distance $\sqrt{2}$, but a standard square grid does not distinguish between these in the spatial convolution kernel. This asymmetry means that convolutional kernels which use rectangular local receptive fields — are implicitly biased towards cardinal directions, potentially amplifying apparent directional heterogeneity in influence results beyond true spatial anisotropy.

In our Φ analysis, the top 10 strongest influence pairs are all between cardinal neighbors (distance = 1 in grid units: e.g., cell 36 and cell 35 are in the same row, adjacent columns). It is therefore possible that some of the directional concentration we observe reflects this cardinal bias rather than purely functional urban structure. Future work should test hexagonal grids (where all 6 neighbors are equidistant at distance 1) or adaptive spatial tessellations aligned to transport network structure, which would allow researchers to disentangle true spatial anisotropy from geometric artefacts.

We also note that at the 8×8 resolution (≈1,500 m cells), our proximity–influence correlation of −0.164 is computed over 4,032 zone pairs spanning grid distances from 1 to ~9.9 units. The slight negative correlation is dominated by the pattern that most low-distance pairs outside the active downtown core carry near-zero Φ, while the high-Φ pairs are at short distances within the core. A partial correlation analysis conditioning on whether zones fall inside or outside the downtown cluster (cells 28, 29, 36, and 37) would be informative for future replication studies.

TEXT S4: DETAILED METHOD

Data. Capital Bikeshare publishes all trip records at https://capitalbikeshare.com/system-data. The raw January 2026 dataset contained 253,418 records. We removed 1,098 trips shorter than one minute — a standard cleaning step applied because sub-60-second transactions almost certainly represent accidental undockings or dock re-locks rather than genuine trips (cf. Gebhart & Noland, 2014) — and 687 trips longer than 12 hours, which almost certainly represent unreturned or lost bicycles. We note that the 12-hour ceiling is deliberately generous relative to Capital Bikeshare's current pricing structure, under which annual members face overage charges after 45 minutes and pay-per-trip users after 30 minutes; a 12-hour record is therefore extremely unlikely to be a legitimate trip under either pricing regime. A sensitivity analysis using a stricter 4-hour ceiling is reported in Supplemental Information Table S1; spatial influence patterns are qualitatively unchanged. After removing an additional 0 records with missing coordinates, 251,633 valid trips remained — 205,329 member trips (81.6%) and 46,304 casual-user trips (18.4%). The study area (38.80°N–39.00°N, 77.15°W–76.90°W) was tiled into an 8×8 regular grid of 64 zones, each approximately 1,500 m per side (Figure 1). Hourly weather records — air temperature, precipitation, wind speed, and relative humidity — were retrieved from the Open-Meteo Historical Weather Archive (https://open-meteo.com/en/docs/historical-weather-api) for the DC centroid (38.9°N, 77.03°W). Because weather observations are hourly, each weather vector was assigned unchanged to both 30-minute steps within its hour; this means the model cannot capture sub-hourly weather fluctuations but introduces no artificial variation.

8×8 grid choice and limitations. The 8×8 resolution (≈1,500 m cells) was adopted to enable direct comparison with Miao et al. (2025), who used the same grid for NYC Citi Bike. We acknowledge three limitations of this choice. First, individual cells at this resolution may encompass functionally distinct neighbourhoods — for example, Georgetown and the National Mall fall in the same zone band — and finer grids would better capture intra-cell heterogeneity, at the cost of greater data sparsity per cell, particularly for casual users in winter. Second, a coarser grid aggregates more trips per cell, which can inflate R^2 by smoothing noise; our $R^2 = 0.816$ should be interpreted in the context of ≈1,500 m zones, not station-level resolution. Third, square grids impose a directional artefact because diagonal neighbours are farther apart than cardinal neighbours, which may amplify apparent directional heterogeneity in influence results. A grid-resolution sensitivity analysis (6×6 and 10×10) is reported in Supplemental Information ***Table S2***.

ConvLSTM models. We trained two independent ConvLSTM models — one for members, one for casual users — to maintain replication comparability with Miao et al. (2025), who also used ConvLSTM, and because ConvLSTM naturally processes gridded spatiotemporal data (Tang et al., 2024). Each model accepted a four-step (2-hour) lookback window of 8×8 pickup and drop-off grids. Three stacked ConvLSTM layers (64 filters, 3×3 convolution kernel) extracted spatiotemporal features; between layers, batch normalization (a technique that stabilizes training by rescaling intermediate outputs) and dropout at rate 0.20 (randomly disabling 20% of connections during training to prevent overfitting) were applied. A parallel dense (fully connected) branch processed seven contextual inputs — temperature, precipitation, wind speed, humidity, hour-of-day, day-of-week, and a weekend flag — expanding from 32 to 64 units, then broadcast spatially across the grid and merged with the ConvLSTM output. A final 1×1 convolution produced pickup and dropoff predictions for all 64 zones simultaneously. Both models used the Adam optimiser (learning rate = 0.001), mean-squared-error loss, and early stopping (training halted when validation loss stopped improving for 10 consecutive epochs, restoring the best weights) on an 80/20 chronological split. We note that while our finding of non-proximity spatial influence may argue for graph-based architectures — such as Graph Attention Networks (He & Shin, 2020), which can model arbitrary zone relationships without the implicit proximity bias of convolution kernels — adopting a different architecture here would confound the replication comparison. Comparing ConvLSTM with graph-based approaches on the same DC data is a natural next step. A simple baseline comparison (historical average and plain LSTM without spatial convolution) is reported in ***Table S3***.

Spatial influence analysis. To quantify how much the demand history of each zone affects the model's predictions for every other zone, we used permutation-based perturbation — a model-agnostic technique that does not require access to model gradients. For each ordered pair of zones (source zone s, target zone t), we selected 30 random test sequences, set the source zone's demand values to zero throughout the lookback window, re-ran the model, and measured the mean absolute change in the target zone's predicted pickups. This change is the spatial influence score $\Phi(s \rightarrow t)$: a high value means that what happens in zone s strongly shapes what the model predicts for zone t. Repeating this for all 64×63 = 4,032 ordered zone pairs yields a 64×64 influence matrix Φ per user type.

We summarize Φ using three statistics:

- *Outward influence* (column sum of Φ — how strongly a zone drives others);
- The coefficient of variation of outward influence scores, which we call the *anisotropy index* (a high value means influence is distributed very unevenly — a few zones dominate);
- Pearson correlation between all pairwise Euclidean grid distances and corresponding Φ values (the *proximity–influence correlation*).

We did not apply formal spatial statistical tests such as Moran's I given the short-paper format, but acknowledge such tests would provide additional confirmation of the patterns reported here.

TEXT S5: MODEL JUSTIFICATION AND TRADEOFFS

This choice involves trade-offs. Large zones may combine functionally different neighborhoods; coarser grids can increase R^2 by smoothing demand; and square cells may accentuate directional patterns (Supplemental Text S3). Sensitivity checks using 6×6 and 10×10 grids show that the main substantive findings remain unchanged (Supplemental Table S2).

We trained two separate ConvLSTM models, one for members and one for casual users, following the architecture used by Miao et al. (2025). ConvLSTM is suitable here for two reasons. First, the main goal is replication and extension of a prior study that used a grid-based deep-learning framework, so keeping the same modeling family

helps isolate the contribution of adding weather and changing the study city. Second, the demand data were aggregated to a regular lattice, making convolution a natural way to learn local spatial patterns and their evolution over time.

That said, a graph-based model could also be appropriate, and in some respects may be more natural for bike-share systems because stations and trips form network relations rather than perfect Euclidean grids (He & Shin, 2020). This is especially relevant given our finding that geographic proximity alone explains little of the estimated influence structure. We therefore do not argue that ConvLSTM is universally superior to graph-based approaches. Rather, we use it as a consistent and interpretable replication framework. In this study, its practical advantages are comparability with Miao et al. (2025), strong predictive performance relative to simpler baselines, and straightforward implementation on gridded inputs. Future work should test whether graph neural networks recover similar non-proximity dependencies when the system is represented directly as an origin-destination or station network.