Forecasting West Nile Virus with Deep Graph Encoders

Trevor Harris

August 6, 2025

University of Connecticut

Collaborators

Left: Ethan Greiffenstein (Incoming PhD student at Texas A&M University)

Right: Rebecca Smith (Associate Professor of Pathobiology at UIUC)

West Nile Virus

- West Nile virus (WNV) is the leading cause of mosquito-borne illness in the United States.
- There is no human vaccine or cure for WNV. The best mechanism for control is surveillance and prevention.
- Illinois Department of Public Health (IDPH) uses an extensive network of mosquito traps to monitor for the presence of WNV.
    - Each day a subset of these traps are tested for the presence of WNV
    - About 15,600 mosquito pools are tested each year

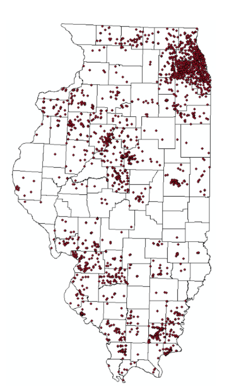


Figure 1: Geographic distribution of traps in Illinois (2018)

Forecasting question

- Early warning systems of WNV attempt to forecast when and where WNV will soon be present
- Accurate short term forecasting can aid mosquito control by helping guide surveillance and providing early warnings of transmission risk to humans
- Individuals in soon to be high-risk areas can take precaution

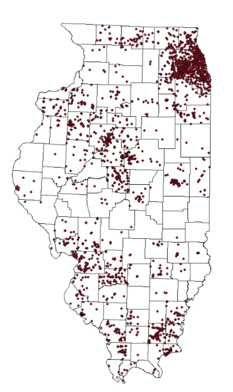


Figure 2: Geographic distribution of traps in Illinois (2018)

Question: Which traps are likely to test positive in the near future?

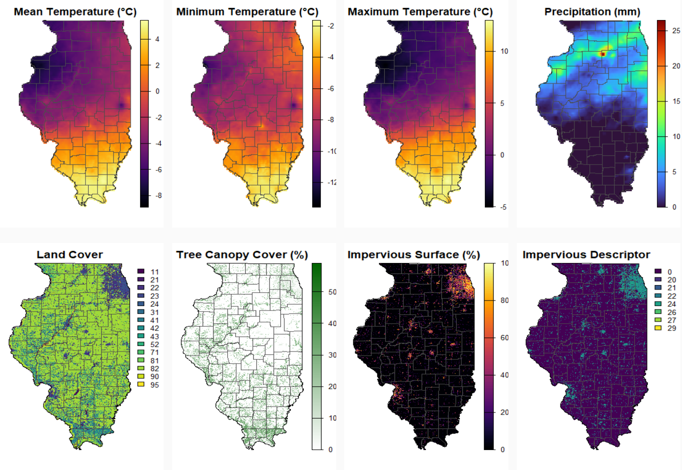
Data - Response

Illinois Department of Public Health (IDPH) surveillance data

- $n = 133,867$ observations from 2008 to 2021.
- Records whether a given trap on a given day is positive (1) or negative (0) for WNV
- Group surveillance results by week $\Rightarrow$ try to predict whether a given trap will test postive *at any time* during the next 1,2,...,7 weeks.
- Dataset variables: lat / lon, time stamp, mosquito count, WNV presence

Does not include any substantial covariate data

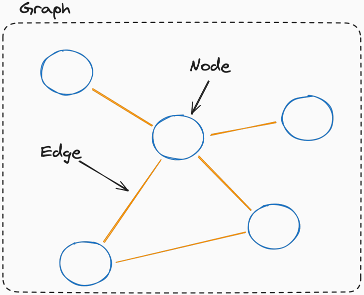
Data - Covariates

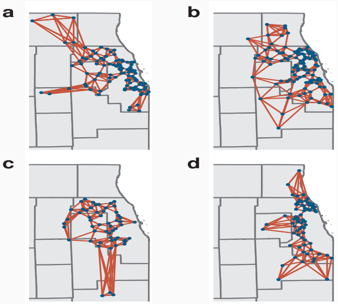


WNV predicted by (lagged) environmental conditions

⇒ How to integrate different **data sources** into our predictions?

Graphs



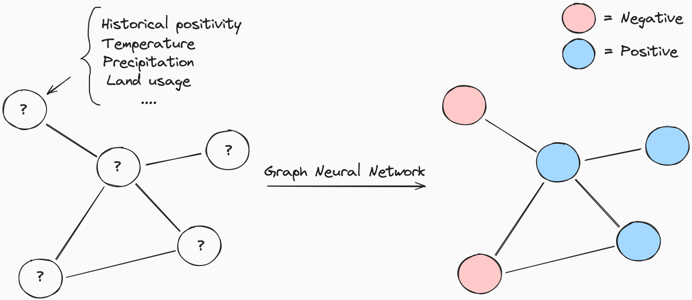


Defn. A graph is a collection of *nodes* and *edges*. Nodes represent entities and edges represent the relationships between them.

$\Rightarrow$ Convert spatial data into spatial graphs. Represent trap locations as *nodes* and connect "nearby" traps with an *edge*

Graph neural networks

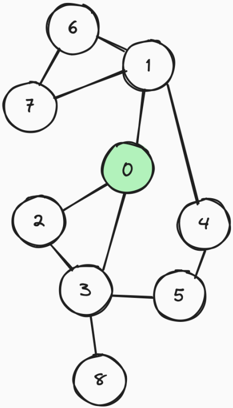
Defn. A graph neural network is a neural network that takes a graph as input and returns a prediction at each node.



GNN – algorithm sketch

1. Process the covariates *at each node* with a neural network
2. Aggregate output with neighbors. Repeat.
3. Finally output probability of being positive

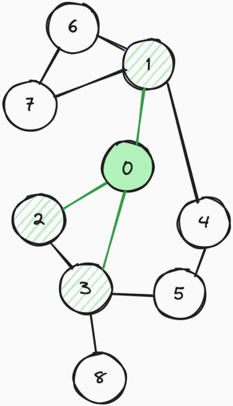
Graph neural networks



1. Feed the covariates X_0 (lagged positivity and weather) through a standard neural network f_θ

$$f_\theta(X_0) = Z_0 \quad \text{(latent representation)}$$

Graph neural networks



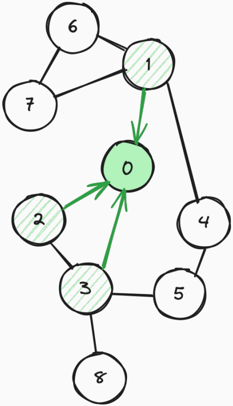
1. Feed the covariates X_0 (lagged positivity and weather) through a standard neural network f_θ

$$f_\theta(X_0) = Z_0 \quad \text{(latent representation)}$$

2. Feed each neighbor's covariates through the same neural network

$$f_\theta(X_1) = Z_1, f_\theta(X_2) = Z_2, f_\theta(X_3) = Z_3$$

Graph neural networks



1. Feed the covariates X_0 (lagged positivity and weather) through a standard neural network f_θ

$$f_\theta(X_0) = Z_0 \quad \text{(latent representation)}$$

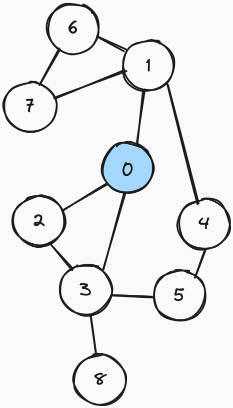
2. Feed each neighbor's covariates through the same neural network

$$f_\theta(X_1) = Z_1, f_\theta(X_2) = Z_2, f_\theta(X_3) = Z_3$$

3. Update the output with weights W

$$Z_0^1 \leftarrow W * \text{concat}([Z_0, \text{Avg}(Z_1, Z_2, Z_3)])$$

Graph neural networks



1. Feed the covariates X_0 (lagged positivity and weather) through a standard neural network f_θ

$$f_\theta(X_0) = Z_0 \quad \text{(latent representation)}$$

2. Feed each neighbor's covariates through the same neural network

$$f_\theta(X_1) = Z_1, f_\theta(X_2) = Z_2, f_\theta(X_3) = Z_3$$

3. Update the output with weights W

$$Z_0^1 \leftarrow W * \text{concat}([Z_0, \text{Avg}(Z_1, Z_2, Z_3)])$$

4. Repeat the above L times for each node to get Z_0^L. Squash to probability

$$p_0 = \frac{1}{1 + \exp(-Z_0^L)}$$

GraphSAGE

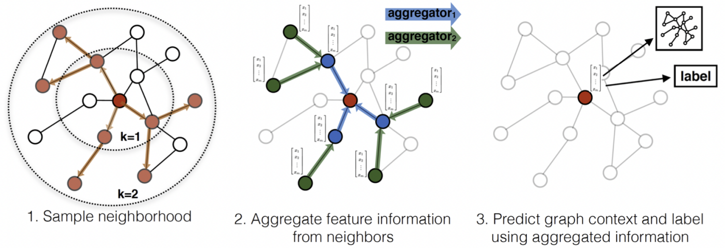


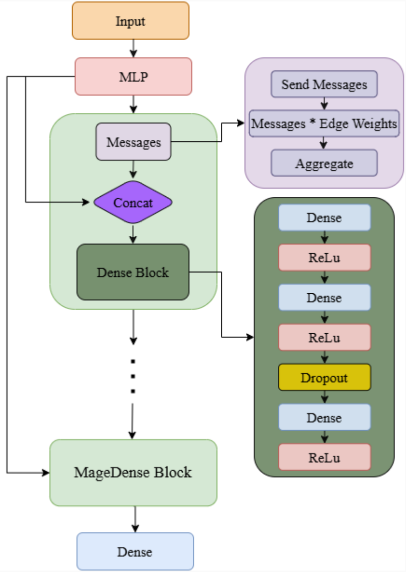
Figure 1: Visual illustration of the GraphSAGE sample and aggregate approach.

In practice we use a specific kind of graph processing layer called GraphSAGE. Exactly the same except...

- Samples the neighboring nodes instead of using all neighboring nodes
- Allows us to train on unlabeled nodes (traps that weren't checked that particular day) - semi-supervised learning

Deep Graph Encoding

- Fundamental problem of GNNs is *prediction collapse*
  - If depth $L \to \infty$, then GNN predicts all nodes to have the same value (over-smoothing)
  - But! depth is key to scaling NNs to large, complex datasets?
- Solution: introduce skip connections
  - Residual GNNs act like ensemble of GNNs of depth $1, 2, .., K$
  - Shallow networks balance deep networks



(Semi) Supervised training

1. **Semi-supervised training**
   - Given graph $G_t = (V_t, E_t)$, i.e. traps V_t with edge set E_t only some of the nodes will be checked (observe y_{v_t} for $v_t \in V_t$).
   - Compute graph embeddings at *all* nodes. Evaluate BCE loss at *checked* nodes.

$$\mathcal{L}(\theta) = -\sum_{v_t \in V_t} (y_{v_t} \log(p_{v_t}) + (1 - y_{v_t}) \log(1 - p_{v_t})) 1(v_t \text{ checked})$$

2. **Supervised training**
   - Given $G_t = (V_t, E_t)$, delete all *unchecked* nodes + edges
   - Compute graph embeddings at *all* remaining nodes. Evaluate BCE loss at *all* remaining nodes.

$$\mathcal{L}(\theta) = -\sum_{v_t \in V_t} (y_{v_t} \log(p_{v_t}) + (1 - y_{v_t}) \log(1 - p_{v_t}))$$

Optimize with stochastic gradient descent + gradient clipping

Numerical Experiments

Model setup (GraphMAGE)

- Model: four GraphMAGE layers (Width = 128), relu activations, Dropout (p = 0.2). Consider supervised (GraphMAGE-SL) and semi-supervised (GraphMAGE-SSL) variant.
- Fit 7 separate models corresponding to 7 different time horizons: week 1, week 2, ..., week 7
- Graph: Traps are connected to their (at most) 10 nearest neighbors within a radius of 50 km.

Baselines: Standard GNN (MageNet baseline), Standard GNN with full residual connections, random forest, logistic regression

Full dataset is split into training (2008 to 2016), validation (2017 to 2018) and test (2019 to 2021) sets.

Evaluation - Overall

F1 Score (↑)								
Model	Week 0	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7
GraphMAGE-SSL	**0.5867**	**0.5216**	0.5106	**0.4894**	**0.4745**	**0.4782**	**0.484**	0.4949
GraphMAGE-SL	0.5848	0.5121	**0.5121**	0.4893	0.4719	0.4689	0.4736	**0.4954**
MageNet Baseline	0.4227	0.3929	0.3886	0.3763	0.3578	0.3733	0.3576	0.3671
MageNet Baseline ResGCN	0.3978	0.3598	0.3283	0.2816	0.2825	0.2411	0.3118	0.3301
GraphMAGE-SL ResGCN	0.4982	0.4696	0.4529	0.4254	0.4134	0.4137	0.4198	0.4571
Random Forest	0.3373	0.1857	0.0968	0.0343	0.0067	0.0001	0.0001	0
Logistic Regression	0.2832	0.2551	0.2417	0.2489	0.236	0.2505	0.2418	0.2266

Table 1: Performance averaged over all nodes when the model has access to entire dataset.

GraphMAGE-SSL and GraphMAGE-SL significantly better WNV classifiers than baseline approaches.

AUC, Brier Scores, sensitivity, specificity, accuracy show similar patterns

Evaluation - Urban to Rural

F1 Score (↑)								
Model	Week 0	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7
GraphMAGE-SSL	0.3438	0.287	0.2688	0.2468	0.2496	0.2421	0.2477	0.2511
GraphMAGE-SL	**0.3451**	**0.2891**	**0.2781**	**0.2538**	**0.2499**	**0.2429**	**0.2475**	**0.2522**
MageNet Baseline	0.266	0.2078	0.209	0.1874	0.1796	0.1871	0.1854	0.1793
MageNet Baseline ResGCN	0.2466	0.1834	0.1819	0.1628	0.1567	0.1638	0.1545	0.1549
GraphMAGE-SL ResGCN	0.3030	0.2523	0.2564	0.2307	0.2306	0.2298	0.2353	0.2166
Random Forest	0.1818	0.0454	0.0014	0	0	0	0	0
Logistic Regression	0.1961	0.1707	0.1621	0.1595	0.1586	0.1564	0.1531	0.1327

Table 2: Performance averaged over all nodes when the model has access to entire dataset.

- Question: Do models trained on rural/suburban (weakly connected) data generalize to urban (sparsely connected)?
- GraphMAGE-SSL and GraphMAGE-SL generalize better then baselines
- Significant room for improvement, very important problem!

Evaluation - Rural to Urban

F1 Score (↑)								
Model	Week 0	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7
GraphMAGE-SSL	0.6365	**0.5936**	**0.5692**	**0.5461**	0.5225	**0.5383**	**0.5507**	**0.5551**
GraphMAGE-SL	**0.6417**	0.5882	0.5607	0.5457	**0.525**	0.5259	0.5353	0.5509
MageNet Baseline	0.5593	0.4549	0.4341	0.4251	0.3935	0.4091	0.4376	0.4265
MageNet Baseline ResGCN	0.4709	0.3588	0.2920	0.3039	0.2857	0.2885	0.2799	0.3057
GraphMAGE-SL ResGCN	0.5448	0.5524	0.5010	0.4886	0.4620	0.4732	0.5078	0.5406
Random Forest	0.3705	0.1976	0.0864	0.0087	0	0	0	0
Logistic Regression	0.2958	0.2808	0.253	0.2777	0.2521	0.2831	0.2679	0.2445

Table 3: Performance averaged over all nodes when the model has access to entire dataset.

- Question: Do models generalize from rural to urban?
- GraphMAGE-SSL and GraphMAGE-SL generalize better then baselines

Entropy Reduction

- CDC wants to know *where* to place new traps
- Calibrate model probabilities and estimate entropy (uncertainty)
- Large differences between columns $\Rightarrow$ add nodes in these areas!

	GraphMAGE-SSL			GraphMAGE-SL		
Variable	All Nodes	Upper 80	Lower 80	All Nodes	Upper 80	Lower 80
Canopy						
Low	0.3083	0.3592	0.3489	0.3094	0.3619	0.3515
Medium	0.3489	0.3879	0.3898	0.3529	0.3925	0.3920
High	0.2778	0.4027	0.3650	0.2873	0.4082	0.3624
Imperviousness						
Low	0.3073	0.3635	0.3532	0.3095	0.3657	0.3554
Medium	0.3406	0.3767	0.3717	0.3407	0.3809	0.3750
High	0.3101	0.3767	0.3565	0.3165	0.3809	0.3611
Land Usage						
Open Water	0.2736	0.3422	0.3270	0.2779	0.3434	0.3301
Open Space Dev.	0.3059	0.3697	0.3598	0.3075	0.3716	0.3611
Low Dev.	0.3298	0.3747	0.3688	0.3317	0.3774	0.3713
Medium Dev.	0.3252	0.3704	0.3635	0.3258	0.3809	0.3750
High Dev.	0.3058	0.3673	0.3526	0.3084	0.3713	0.3553
Deciduous Forest	0.3240	0.3724	0.3597	0.3248	0.3744	0.3599
Grasslands Herb.	0.2793	0.3414	0.3249	0.2800	0.3417	0.3243
Pasture Hay	0.2902	0.3712	0.3455	0.2961	0.3720	0.3487
Cultivated Crops	0.2668	0.3448	0.3192	0.2684	0.3485	0.3232
Woody Wetlands	0.3243	0.3646	0.3609	0.3226	0.3681	0.3611
Roads						
Primary Road	0.2736	0.3422	0.3270	0.2779	0.3434	0.3301
Secondary Road	0.3059	0.3697	0.3598	0.3075	0.3716	0.3611
Tertiary Road	0.3298	0.3747	0.3688	0.3317	0.3774	0.3713
Non Road	0.3252	0.3704	0.3635	0.3258	0.3734	0.3663

Table 4: Entropy by land usage type.

Summary

- Short lead forecasting is important for mosquito control and early warning about WNV transmission risk
- We propose a new approach based on Deep Graph Neural Networks (GNNs) for predicting if traps will test positive in the next $1-7$ weeks.
  - Highly flexible and scalable to very large datasets
  - GNNs show promising results in many other application areas
  - Directly accounts for spatial dependence through the imposed spatial graph
- New architecture vastly out scales previous GNN approaches (and non-spatial, linear approaches)
- Current approaches that do not account for spatial dependence are unable to reach a similar level of forecasting skill

Thanks for listening!