Heterogeneous Peer Effects in the Linear Threshold Model

Appendix

Individual threshold estimation for LTM

The implication of mapping the node threshold estimation to a trigger-based heterogeneous treatment effect estimation problem is that an accurate trigger correctly estimates the node threshold.

Theorem 1. *For any node $v \in \boldsymbol{V}$, let θ_v be v's true threshold. Then $\hat{\theta}_v$ that maximizes the CAPE with a trigger in eq (12):*

$$\underset{\hat{\theta}_v}{\operatorname{argmax}} \; E[\mathcal{Y}(I_v^t \geq \hat{\theta}_v) - \mathcal{Y}(I_v^t < \hat{\theta}_v) \mid \mathbf{X}_v, Z_v], \quad (1)$$

provides the best estimate of the node threshold, θ_v.

Proof. Suppose $|N(v)| = n$ and w_{uv} be an arbitrary weight such that $\sum_u w_{uv} = 1$. Let the true threshold of node v be θ_v. Define $\mathbf{A_v} = \mathbf{a_i}$ to be an assignment of activations of neighbors of v and $W_v(\alpha_i)$ to be the outcome for the activated set $\mathbf{a_i}$:

$$W_v(\alpha) = \begin{cases} 0 & \text{if } I_v(\mathbf{a_i}) < \theta_v, \\ 1 & \text{if } I_v(\mathbf{a_i}) \geq \theta_v. \end{cases} \quad (2)$$

Where I_v is the activation influence. Let the set of potential outcomes, $\mathcal{Y}_v$, below and above the estimated trigger $\hat{\theta_v}$ be:

$$\mathcal{Y}_v(I_v < \hat{\theta}_v) = \{W_v(\mathbf{a_i}) \mid I_v(\mathbf{a_i}) < \hat{\theta_v}\} \quad (3)$$

$$\mathcal{Y}_v(I_v \geq \hat{\theta}_v) = \{W_v(\mathbf{a_i}) \mid I_v(\mathbf{a_i}) \geq \hat{\theta}_v\}. \quad (4)$$

Let N_0 and N_1 be the size of the sets $\mathcal{Y}_v(I_v < \hat{\theta}_v)$ and $\mathcal{Y}_v(I_v \geq \hat{\theta}_v)$, respectively. Define the set of outcomes when node v is not activated and activated with the true threshold as $\mathcal{Z}_v(0)$ and $\mathcal{Z}_v(1)$:

$$\mathcal{Z}_v(0) = \{W_v(\mathbf{a_i}) \mid I_v(\mathbf{a_i}) < \theta_v\}, \quad (5)$$

$$\mathcal{Z}_v(1) = \{W_v(\mathbf{a_i}) \mid I_v(\mathbf{a_i}) \geq \theta_v\}. \quad (6)$$

Let $N_{\mathcal{Z}_0}$ and $N_{\mathcal{Z}_1}$ be the cardinalities of set $\mathcal{Z}_v(0)$ and $\mathcal{Z}_v(1)$, respectively. The difference between $\mathcal{Y}_v$ and $\mathcal{Z}_v$ is that $\mathcal{Y}_v$ contains the true potential activations based on the *estimated* threshold and $\mathcal{Z}_v$ contains the true potential activations based on the *true* threshold. We define expectation over the sets $\mathcal{Y}_v$ and $\mathcal{Z}_v$ as the mean over the set. There are three cases for the estimated threshold, $\hat{\theta}_v$:

Case 1 ($\hat{\theta}_v < \theta_v$): We compute the expected outcomes below and above the estimated trigger $\hat{\theta}_v$:

$$E[\mathcal{Y}_v(I_v < \hat{\theta}_v)] = E[\{W_v(\mathbf{a_i}) \mid I_v(\mathbf{a_i}) < \hat{\theta}_v\}] = \frac{1}{N_0} \sum_i W_v(\mathbf{a_i}) = 0, \quad (7)$$

since there are no times when W_v will be 1 (activated) since all outcomes below the estimated threshold $\hat{\theta}_v$ are also below the true threshold θ_v. The expected value above the estimated threshold is:

$$E[\mathcal{Y}_v(I_v \geq \hat{\theta}_v)] = E[\{W_v(\mathbf{a_i}) \mid I_v(\mathbf{a_i}) \geq \hat{\theta}_v\}] = \frac{1}{N_1} \sum_i W_v(\mathbf{a_i}) = \frac{1}{N_1} \sum_i W_v(\mathbf{a_i}) + \frac{1}{N_1} \sum_j W_v(\mathbf{a_j}) = \frac{1}{N_1} \cdot N_{\mathcal{Z}_1} = \frac{N_{\mathcal{Z}_1}}{N_1}. \quad (8)$$

The effect is the difference above and below the estimated trigger:

$$E[\mathcal{Y}_v(I_v \geq \hat{\theta_v}) - \mathcal{Y}_v(I_v < \hat{\theta_v})] = \frac{N_{\mathcal{Z}_1}}{N_1} - 0 = \frac{N_{\mathcal{Z}_1}}{N_1} < 1. \quad (9)$$

Note that $N_{\mathcal{Z}_1} < N_1$ since N_1 has cases above the estimated threshold, which is smaller than the true threshold.

Case 2 ($\hat{\theta}_v > \theta_v$):

$$E[\mathcal{Y}_v(I_v < \hat{\theta}_v)] = E[\{W_v(\mathbf{a_i}) \mid I_v(\mathbf{a_i}) < \hat{\theta}_v\}] = \frac{1}{N_0} \sum_i W_v(\mathbf{a_i}) + \frac{1}{N_0} \sum_j W_v(\mathbf{a_i}) = 0 + \frac{1}{N_0} \cdot (N_0 - N_{\mathcal{Z}_0}) = \frac{N_0 - N_{\mathcal{Z}_0}}{N_0}. \quad (10)$$

$$E[\mathcal{Y}_v(I_v \geq \hat{\theta}_v)] = E[\{W_v(\mathbf{a_i}) \mid I_v(\mathbf{a_i}) \geq \hat{\theta}_v\}] = \frac{1}{N_1} \cdot N_1 = 1. \quad (11)$$

$$E[\mathcal{Y}_v(I_v \geq \hat{\theta_v}) - \mathcal{Y}_v(I_v < \hat{\theta_v})] = 1 - \frac{N_0 - N_{\mathcal{Z}_0}}{N_0} < 1. \quad (12)$$

Case 3 ($\hat{\theta}_v = \theta_v$):

$$E[\mathcal{Y}_v(I_v < \hat{\theta}_v)] = E[\{W_v(\mathbf{a_i}) \mid I_v(\mathbf{a_i}) < \hat{\theta}_v\}] = \frac{1}{N_0}\sum_i W_v(\mathbf{a_i}) = 0. \quad (13)$$

$$E[\mathcal{Y}_v(I_v \geq \hat{\theta}_v)] = E[\{W_v(\mathbf{a_i}) \mid I_v(\mathbf{a_i}) \geq \hat{\theta}_v\}] = \frac{1}{N_1}\sum_i W_v(\mathbf{a_i}) = 1. \quad (14)$$

$$E[\mathcal{Y}_v(I_v \geq \hat{\theta_v}) - \mathcal{Y}_v(I_v < \hat{\theta_v})] = 1 - 0 = 1. \quad (15)$$

The maximum causal effect only occurs when $\hat{\theta}_v = \theta_v$. Therefore if we can estimate the trigger that maximizes CAPE we find the true node threshold. As a result, we can use any trigger-based HTE estimation method to estimate node thresholds. □

Threshold estimation algorithms

Trigger-based causal trees Here we describe the Causal Tree (CT-HV) algorithm proposed in (Tran and Zheleva 2019). Let $\mathbf{X}^\ell$ be the features in partition ℓ, which represents a node in the tree which contains N_ℓ samples, and let $\hat{\mu}_1(\ell)$ and $\hat{\mu}_0(\ell)$ be the mean of outcomes when treated and non-treated. The estimate of CATE of any partition is $\hat{\tau_c}(\mathbf{X}^\ell) = \hat{\mu}_1(\ell) - \hat{\mu}_0(\ell)$. Given a partition ℓ that needs to be partitioned further into two children ℓ_1, ℓ_2, the causal tree algorithm finds the split on features that maximizes the weighted CATE in each child:

$$\max_{\ell_1,\ell_2} N_{\ell_1} \cdot \hat{\tau_c}(\mathbf{X}^{\ell_1}) + N_{\ell_2} \cdot \hat{\tau_c}(\mathbf{X}^{\ell_2}). \quad (16)$$

We refer to the two quantities $N_{\ell_1} \cdot \hat{\tau_c}(\mathbf{X}^{\ell_1})$ and $N_{\ell_2} \cdot \hat{\tau_c}(\mathbf{X}^{\ell_2})$ as partition measures for ℓ_1 and ℓ_2. In our work, we adapt the CT-HV algorithm (Tran and Zheleva 2019) for the problem of node threshold estimation.

In order to learn triggers, an additional search is done at each split to find the trigger that maximizes the effect estimation in each split. Let $F(\ell)$ represent the partition measure for CT-HV and let $\hat{\theta}_v$ be some estimated trigger for partition ℓ. Define $M_1(\ell, \hat{\theta}_v)$ and $M_0(\ell, \hat{\theta}_v)$ to be the expected mean outcomes when above and below the trigger $\hat{\theta}_v$, and $F(\ell, \hat{\theta}_v)$ be the partition measure with trigger $\hat{\theta}_v$: $F(\ell, \hat{\theta}_v) = M_1(\ell, \hat{\theta}_v) - M_0(\ell, \hat{\theta}_v) = E[\mathcal{Y}_\ell(I \geq \hat{\theta}_v) - \mathcal{Y}_\ell(I < \hat{\theta}_v)]$. Then to find a trigger in partition ℓ, we find the trigger that maximizes the partition measure: $F(\ell, \hat{\theta}_v)$. This results in a unique trigger in each partition.

ST-Learner

Pseudocode is provided for ST-Learner in Algorithm 1. Both the training and prediction subroutine are included. In the train subroutine, we train a base learner, f to predict the outcome Y using node features $\mathbf{X}$ and the activation influence I. To predict a trigger for a new test example, we compute and score the predicted outcome for all treatment values found in the dataset. Then, for each potential trigger to consider, we compute an average above and below that trigger, and find the trigger that maximizes the difference in outcomes. The additional complexity added by the search for triggers is on the order of potential treatment values $|B|$. For example, suppose a base learner of Linear Regression requires $O(d)$ time for prediction, then ST-Learner with Linear Regression will take $O(d \cdot |B|)$ time.

Algorithm 1 ST-Learner

Input: Training set $(\mathbf{X}, Y, I)$, test example $\mathbf{x}^{te}$, base learner f, possible treatments $\boldsymbol{\beta}$, possible triggers Θ
Output: The causal effect estimate for $\mathbf{x}^{te}$

function TRAIN($\mathbf{X}, Y, I$)
  Fit f to predict Y, f: $(\mathbf{X}, I) \rightarrow Y$
  $\implies f(\mathbf{X}, I) = E[Y \mid \mathbf{X}, I]$
function PREDICT($\mathbf{x}^{te}$)
  for each β_i in $\boldsymbol{\beta}$ **do**
    Compute and store $f(\mathbf{x}^{te}, \beta_i)$
  max_trigger $= 0$, max_effect $= 0$
  for each r_i in Θ **do**
    Compute $t_1 = \frac{1}{|\Theta_i^1|}\sum_{\beta_k \geq r_i} f(\mathbf{x}^{te}, \beta_k)$
    Compute $t_0 = \frac{1}{|\Theta_i^0|}\sum_{\beta_j < r_i} f(\mathbf{x}^{te}, \beta_j)$
    $e = t_1 - t_0$
    if $e >$ max_effect
      max_trigger $= r_i$, max_effect $= e$
  return max_trigger, max_effect

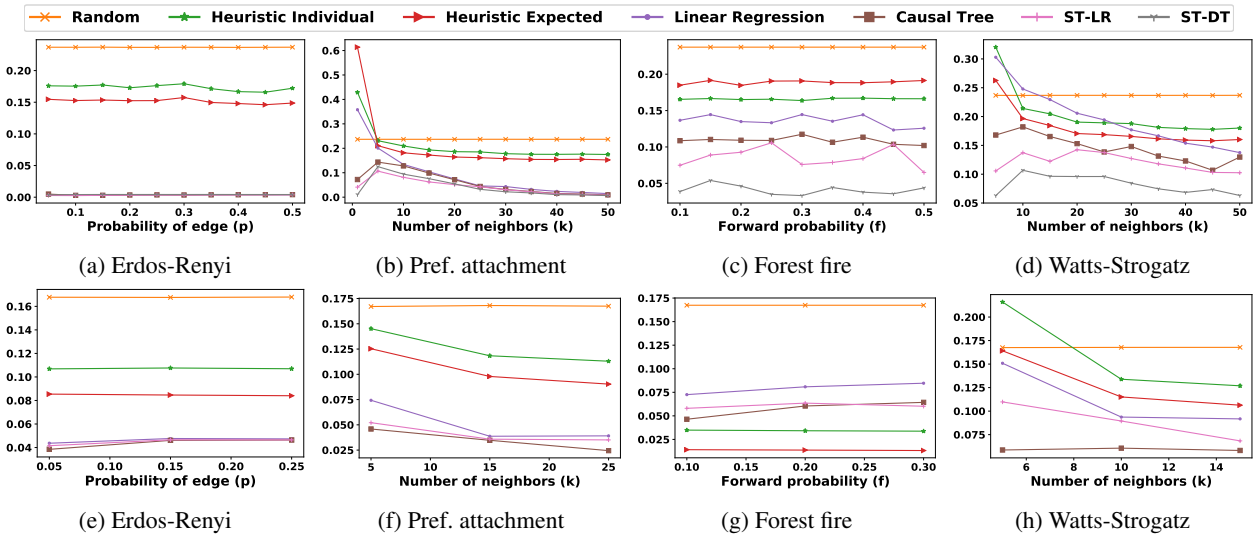
Additional results

Task 1: node threshold prediction Figure 1 shows the change in MSE in the *Linear* and *Quadrant* setup. From the results, we see that ST-DT in general performs better on the nonlinear threshold in the *Quadrant* dataset, compared to the linearly generated threshold.

Task 3: diffusion size prediction

Figure 2 shows additional figures for the real-world datasets. Each figure starts at a different network snapshot, where a snapshot is the current structure and activations at time t. For example, Figure 2b learns on data starting at the snapshot at the end of March 2016. A later snapshot means there are more activations in the dataset, and thus more training data. As the training data size increases, our methods can achieve better diffusion size prediction.

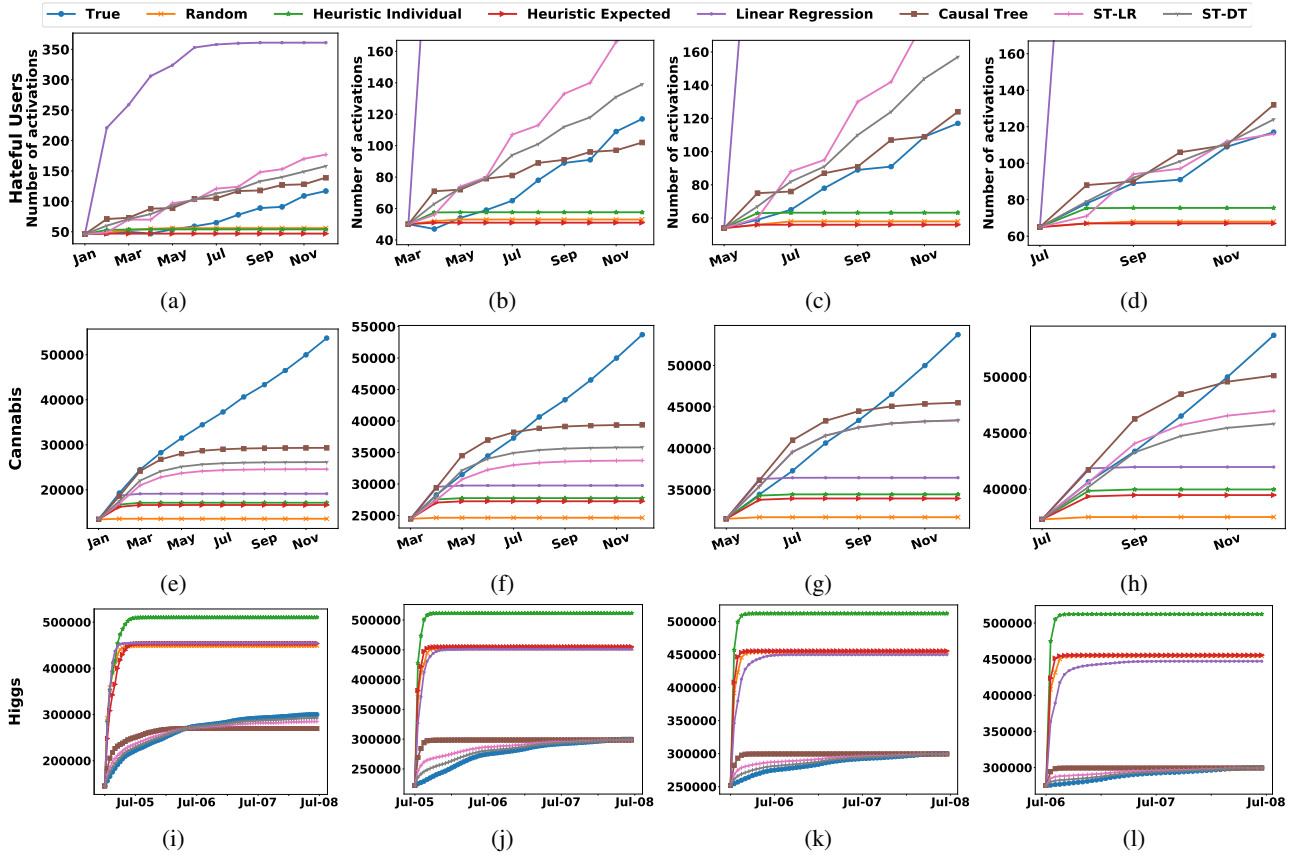
Hateful Users dataset. Figures 2a-2d show the diffusion predictions for the Hateful Users dataset from Jan 2016 to Dec 2016, with varying starting points (Jan, Mar, May, Jul). We show diffusion simulations at four snapshots before Dec 2016. Our models initially overestimate the diffusion size prediction The first thing we notice is that our models initially overestimates the reach the diffusion prediction, but is able to predict close to the final diffusion amount. Additionally, we see that as we get more time steps, our models obtains more accurate reach estimates. For example, in Figures 2a and 2b, where we start with information in January and March, our models overestimates the prediction in the beginning. With more information, the reach estimates are better, as in Figures 2c (starting in May), and 2d (starting in July). In the 2017 dataset, we notice the same trends as in the 2016 dataset: our models overestimates the reach prediction



(a) Erdos-Renyi (b) Pref. attachment (c) Forest fire (d) Watts-Strogatz

(e) Erdos-Renyi (f) Pref. attachment (g) Forest fire (h) Watts-Strogatz

Figure 1: MSE of threshold prediction for the *Linear* (top row) and *Quadrant* (bottom row) setups over different parameters.



(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

Figure 2: Comparison of diffusion size prediction on three real world datasets. Our models have the closest estimation of reach over longer time periods whereas the baselines incorrectly predict diffusion saturation in the early stages.

in the beginning, but predicts close to the final true amount, so we omit the plots.

Cannabis dataset. Figures 2e-2h show results for threshold estimation on the Cannabis dataset from Jan 2017 to Dec 2017. Contrast to the Hateful Users dataset, all models predict diffusion that saturate after a number of time steps. Our models predict saturation closer to the final diffusion amount with more training data.

Higgs dataset. Figure 2i shows the diffusion size predictions for the Higgs dataset. Specifically, we start at noon, July 4th and increase the starting snapshots by 12 hours each time. We see that the ST-DT performs slightly better than Causal Tree and both outperform the baselines by a significant margin. Additionally, the baselines significantly overestimate the reach predictions.

References

Tran, C.; and Zheleva, E. 2019. Learning Triggers for Heterogeneous Treatment Effects. In *Proceedings of the AAAI Conference on Artificial Intelligence*.