Bayesian Optimization of Functions over Node Subsets in Graphs

Huidong Liang   **Xingchen Wan***   **Xiaowen Dong**
Department of Engineering Science, University of Oxford
{huidong.liang,xiaowen.dong}@eng.ox.ac.uk, xingchenw@google.com

Abstract

We address the problem of optimizing over functions defined on node subsets in a graph. The optimization of such functions is often a non-trivial task given their combinatorial, black-box and expensive-to-evaluate nature. Although various algorithms have been introduced in the literature, most are either task-specific or computationally inefficient and only utilize information about the graph structure without considering the characteristics of the function. To address these limitations, we utilize Bayesian Optimization (BO), a sample-efficient black-box solver, and propose a novel framework for combinatorial optimization on graphs. More specifically, we map each k-node subset in the original graph to a node in a new combinatorial graph and adopt a local modeling approach to efficiently traverse the latter graph by progressively sampling its subgraphs using a recursive algorithm. Extensive experiments under both synthetic and real-world setups demonstrate the effectiveness of the proposed BO framework on various types of graphs and optimization tasks, where its behavior is analyzed in detail with ablation studies. The experiment code can be found at github.com/LeonResearch/GraphComBO.

1   Introduction

In the analysis and optimization of transportation, social, and epidemiological networks, one is often interested in finding a node subset that leads to the maximization of a utility. For example, incentivizing an initial set of users in a social network such that it leads to the maximum adoption of certain products; protecting a set of key individuals in an epidemiological contact network such that it maximally slows down the transmission of disease; identifying the most vulnerable junctions in a power grid or a road network such that interventions can be made to improve the resilience of these infrastructure networks.

The scenarios described above can be mathematically formulated as optimizing over a utility function defined on node subsets in a graph, which is a non-trivial task for several reasons. First, most conventional optimization algorithms are designed for continuous space and are hence not directly applicable to functions defined on discrete domains such as graphs. Second, optimizing over a k-node subset leads to a large search space even for moderate graphs, which are not even fully observable in certain scenarios (e.g. offline social networks). Finally, the objective functions are usually black-box and expensive to evaluate in many applications, such as the outcome of a diffusion process on the network [56] or the output of a graph neural network [50], making sample-efficient queries a necessary requirement.

Assuming the graph structure is fully available, the optimization task described above shares similarities with those encountered in the literature on network-based diffusion. In that literature, greedy

---

*Now at Google.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

algorithms [28, 32, 7] have been widely used to select a subset of nodes that maximizes a utility function, for example in the context of influence maximization [48] or source identification [25]. However, as the underlying functions often require calculating expectations over a large number of simulations (e.g. the expected number of eventual infections from an epidemic process), such algorithms often become extremely time-consuming as the evaluation time for each diffusion process increases [2]. To relieve the inefficiency in computation, proxy-based methods, such as PageRank [6], generalized random walks [10], and DomiRank [17], are often used in practice to rank the importance of nodes. However, such methods completely ignore the underlying function and require full knowledge of the graph structure beforehand. Finally, most methods mentioned above are task-specific, and the one designed for a specific diffusion process usually does not generalize well to another.

In this paper, we consider the challenging optimization setting for black-box functions on node subsets, where the underlying graph structure is not fully observable and can only be incrementally revealed by queries on the fly. To facilitate this setting, we propose a novel strategy to conduct the search in a combinatorial graph space (termed "combo-graph") in which each node corresponds to a k-node subset in the original (unknown) graph. The original problem is thus turned into optimization over a function on the combo-graph, where each node value is the utility of the corresponding subset. Traditional graph-traversing methods, such as breadth-first search (BFS) or depth-first search (DFS), may not work well in this case due to the exceedingly large search space and their lack of capability to exploit the behavior of the underlying function. Bayesian optimization (BO), a sample-efficient black-box solver for optimizing expensive functions via surrogate modeling of its behavior, presents an appealing alternative.

Contributions. We propose a novel Bayesian optimization framework for optimizing black-box functions defined on node subsets in a generic and potentially unknown graph. Our main contributions are as follows. To the best of our knowledge, this is the first time BO has been applied to such a challenging optimization setting. Our framework consists of constructing the aforementioned combo-graph, and traversing this combo-graph by moving around a combo-subgraph sampled by a recursive algorithm. Notably, the proposed framework is function-agnostic and applies to any expensive combinatorial optimization problem on graphs. We validate the proposed framework on various graphs with different underlying functions under both synthetic and real-world settings, and demonstrate its superiority over a number of baselines. We further analyze its behavior with detailed ablation studies. Overall, this work opens new paths of research for important optimization problems in network-based settings with real-world implications.

2 Preliminaries

BO [38, 20] is a gradient-free optimization algorithm that aims to find the global optimal point x^* of a back-box function $f : \mathcal{X} \to \mathbb{R}$ over the search space $\mathcal{X}$, which, in the case of maximization, can be written as $x^* = \arg\max_{x \in \mathcal{X}} f(x)$. To efficiently search for the optimum of expensive-to-evaluate functions, BO first builds a *surrogate model* based on existing observations to predict the function values and their uncertainties over the search space $\mathcal{X}$, then utilizes an *acquisition function* to decide the next location for evaluation.

Surrogate model. One of the most common surrogates used in BO literature [44] is the *Gaussian Processes* model: $f(x) \sim \mathcal{GP}(m(x), k(x, x'))$, in which $m(x)$ is the mean function (often set to a constant $\mathbf{0}$ vector) and $k(x, x')$ is a pre-specified covariance function that measures the similarity between data point pairs. With a training set $\mathcal{D}_t = \{\boldsymbol{x}_{1:t}, \boldsymbol{y}_{1:t}\}$ of t observations, the posterior distribution of $f(x_{t+1})$ for a new location x_{t+1} can be analytically computed from the Gaussian conditioning rule, where the mean is given by $\mu(x_{t+1}|\mathcal{D}_t) = \mathbf{k}(x_{t+1}, \mathbf{X}_{1:t})\mathbf{K}_{1:t}^{-1}\mathbf{y}_{1:t}$ with covariance $k(x_{t+1}, x'_{t+1}|\mathcal{D}_t) = k(x_{t+1}, x'_{t+1}) - \mathbf{k}(x_{t+1}, \mathbf{X}_{1:t})\mathbf{K}_{1:t}^{-1}\mathbf{k}(\mathbf{X}_{1:t}, x'_{t+1})$. Note that the computational cost for $\mathbf{K}_{1:t}^{-1}$ is at $\mathcal{O}(t^3)$, which largely restricts the efficiency of $\mathcal{GP}$ when training on large datasets and therefore often requires a local modeling approach.

Acquisition function. Based on the predictive posterior distribution, an acquisition function will be applied to balance the exploration-exploitation trade-off via optimizing under uncertainty. For example, the *Expected Improvement* [37, 26], defined as $\mathrm{EI}_{1:t}(x') = \mathbb{E}\big[[f(x') - f(x^*_{1:t})]^+\big]$ with $[\alpha]^+ = \max(\alpha, 0)$ and $x^*_t = \arg\max_{x_i \in x_{1:t}} f(x_i)$, measures the expected improvement based on the current best query. Then, the next query location is chosen as $x_{t+1} = \arg\max_{x' \in \mathcal{X} \setminus \{x_i\}_{i=1}^t} \mathrm{EI}_{1:t}(x')$,

2

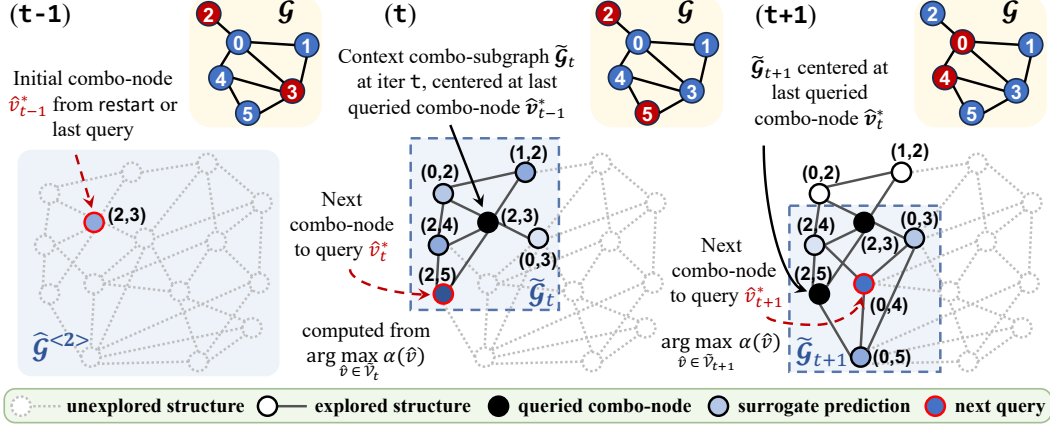



Figure 1: Demonstration of how the proposed framework traverses the combinatorial graph $\hat{\mathcal{G}}^{<k>}$ introduced in §3.1 with an exemplar original graph $\mathcal{G}$ of 6 nodes and a subset size of $k = 2$. At iteration t, we first construct a local combo-subgraph $\tilde{\mathcal{G}}_t = \{\tilde{\mathcal{V}}_t, \tilde{\mathcal{E}}_t\}$ of size Q=6 using Algorithm 1 (§3.1), which is centred at combo-node $\hat{v}_{t-1}^*$ from last iteration t-1 or initialization. Next, a $\mathcal{GP}$ surrogate is fitted on $\tilde{\mathcal{G}}_t$ with queried combo-nodes inside $\tilde{\mathcal{G}}_t$ being the training set. The next query location is then selected as the combo-node that maximizes the acquisition function $\hat{v}_t^* = \arg\max_{\hat{v} \in \tilde{\mathcal{V}}_t} \alpha(\hat{v})$. If queried values $f(\hat{v}_t^*) \geqslant f(\hat{v}_{t-1}^*)$, the next combo-subgraph $\tilde{\mathcal{G}}_{t+1}$ will be re-sampled at a new center $\hat{v}_t^*$, or otherwise remain the same. Finally, we repeat the previous process to obtain a new query location for the next iteration t+1, and the search continues until stopping criteria are triggered.

and the result $\{x_{t+1}, y_{t+1}\}$ will be appended to the visited set $\mathcal{D}_t$. The algorithm will repeat these steps until the stopping criteria are triggered at a certain iteration T, and we report $x_T^* = \arg\max_{x_i \in x_{1:T}} f(x_i)$ as the final result.

3 BO of Functions over Node Subsets in Graphs

Settings and challenges. Following the notations in §2, we formally introduce the proposed Bayesian optimization framework for black-box functions over node subsets in graphs, termed ***GraphComBO***. The goal of the problem is to find the global optimal k-node subset $\mathcal{S}^*$ of a black-box function $f(\mathcal{S})$ over the search space of all possible k-node subsets $\binom{\mathcal{V}}{k}$ on a generic graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, which, in the case of maximization, can be expressed as $\mathcal{S}^* = \arg\max_{\mathcal{S} \in \binom{\mathcal{V}}{k}} f(\mathcal{S})$. Under noisy settings, we may only observe $y = f(x) + \epsilon$, with $\epsilon \sim N(0, \sigma_\epsilon^2)$ being the noise term. For simplicity, we focus on *undirected* and *unweighted* graphs where the adjacency matrix $\mathbf{A}$ is symmetric and contains binary elements. As f is often expensive to evaluate in practice, we wish to optimize the objective in a query-efficient manner within a limited number of evaluations T, and report the best configuration among them as the final solution: $\mathcal{S}_T^* = \arg\max_{\mathcal{S}_i \in \{\mathcal{S}_i\}_{i=1}^T} f(\mathcal{S}_i)$.

Despite BO's appeals in optimizing such functions, we observe the following challenges when designing effective algorithms for combinatorial problems on graphs:

1. **Structural combinatorial space.** Unlike classical combinatorial optimization in the discrete space, the combination of nodes (a node subset) inherits structural information from the underlying graph, which needs to be properly encoded into the combinatorial search space. In addition, an appropriate similarity measure between node-subset pairs is also required to capture such inherent structural information when building the surrogate model.

2. **Imperfect knowledge of graph structures.** As the complete structure of real-world graphs may be expensive or even impossible to acquire (e.g. a gradually evolving social network), any prospectus optimization algorithm needs to handle the situation where the graph structure is only revealed incrementally.

3. **Local approach while combining distant nodes.** As the massive size $\binom{|\mathcal{V}|}{k}$ of the combinatorial space often makes global optimization unattainable, an effective local modeling approach is

3

needed to efficiently traverse the graph. However, as the optimal subset usually consists of nodes that are far away from each other (e.g., the optimal locations of hospitals in a city network), it is critical to maintain the flexibility of selecting distant nodes when considering a local context.

In the following sections, we will discuss how the proposed GraphComBO addresses these challenges, where an overview of the framework can be found in Figure 1.

3.1 The Combinatorial Graph for Node Subsets

Inspired by the graph Cartesian product that projects multiple "subgraphs" into a combinatorial graph, we introduce a combinatorial graph (denoted as ***combo-graph***) tailored for node subsets on a single generic graph with an intuitive example demonstrated in Figure 2.

Definition 3.1. The combinatorial operation for k-node subsets in an underlying graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is given by:

$$\hat{\mathcal{G}}^{<k>} = \square_{i=1}^{k} \mathcal{G}, \tag{1}$$

which leads to a combo-graph $\hat{\mathcal{G}}^{<k>} = \{\hat{\mathcal{V}}, \hat{\mathcal{E}}\}$ of size $|\hat{\mathcal{V}}| = \binom{|\mathcal{V}|}{k}$ with each combo-node $\hat{v}_i = (v_i^{(1)}, v_i^{(2)}, ..., v_i^{(k)}) \in \hat{\mathcal{V}}$ being a k-node subset from the underlying graph $\mathcal{G}$ without replacement. The combo-edges $\hat{\mathcal{E}}$ in the combo-graph are defined in the following way: assume $\hat{v}_1 = (v_1^{(1)}, v_1^{(2)}, ..., v_1^{(k)})$ and $\hat{v}_2 = (v_2^{(1)}, v_2^{(2)}, ..., v_2^{(k)})$ are two arbitrary combo-nodes in the combo-graph $\mathcal{G}^{<k>}$, then $(\hat{v}_1, \hat{v}_2) \in \hat{\mathcal{E}}$ iff $\exists j$ such that $\forall i \neq j$, $v_1^{(i)} = v_2^{(i)}$ and $(v_1^{(j)}, v_2^{(j)}) \in \mathcal{E}$.

Intuitively, this means that in the combo-graph, two combo-nodes are adjacent if and only if they have exactly 1 element (i.e. node from the original graph) in difference and the two different elements are neighbors in the original graph. Note that as k shrinks to 1, the combo-graph reduces to the underlying graph.

Nevertheless, as the combo-graph size $\binom{|\mathcal{V}|}{k}$ is often too large in practice, building the surrogate and making predictions at a global scale is usually unrealistic. A sensible alternative would

Figure 2: Illustration of a combinatorial graph $\hat{\mathcal{G}}^{<2>}$ constructed by the recursive combo-subgraph sampling (Algorithm 1).

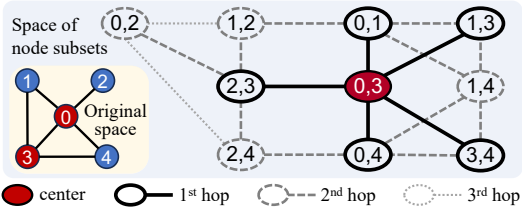
be adopting a commonly used local modeling approach [19] and then gradually moving around the "window" guided by the surrogate predictions. Unlike classical continuous space, constructing local regions on the combo-graph is not straightforward. Next, we will discuss two properties of the proposed combo-graph, which enable us to practically employ local modeling by sampling subgraphs for tractable optimization. Reads are also referred to §D for proofs of the following lemmas.

Lemma 3.2. *In the proposed combo-graph, at most ℓ elements in the subset will be changed between any two combo-nodes that are ℓ-hop away.*

This implies that when considering an ℓ-hop ego-subgraph centered at an arbitrary combo-node $\hat{v}$ on the combo-graph, we are effectively exploring the ℓ-hop neighbors of elements in $\hat{v}$ in the original graph. Since such operation requires no prior knowledge of the other part of the original graph, we are then able to gradually reveal its structure by moving around the focal combo-node, and hence handling the situation of optimizing over node subsets on an incomplete or even unknown graph.

Lemma 3.3. *The degree of combo-node $\hat{v}_i$ increases linearly with k and is maximized by the subset of nodes with top k degrees:* $\deg(\hat{v}_i) = \sum_{j=1}^{k} |\mathcal{N}(v_i^{(j)}) \backslash \{v_i^{(j')}\}_{j' \neq j}^{k}|.$

Therefore, the size of the above ego-subgraph only needs to increase linearly with k to cover the first hop combo-neighbors. These two properties together make the construction of local combo-subgraphs feasible, and we introduce a sampling algorithm in the next section that recursively finds combo-nodes and combo-edges for a combo-subgraph (denoted as $\tilde{\mathcal{G}}$) given a focal combo-node.

Recursive combo-subgraph sampling. As illustrated in Algorithm 1 and Figure 2, our goal is to construct an ego-subgraph $\tilde{\mathcal{G}}$ of size Q from the underlying graph $\mathcal{G}$, centered at a given combo-node

4

$\hat{v}^*$ with maximum hop $\ell_{\max}$. The algorithm initializes $\tilde{\mathcal{G}} = \{\tilde{\mathcal{V}}, \tilde{\mathcal{E}}\}$ with only $\hat{v}^*$ and then loop through neighbors of each element node $v \in \hat{v}^*$. If a neighbor $\mathcal{N}_i(v)$ of v is not in $\hat{v}^*$ (i.e. ensuring no repetition in the subset), a new combo-node $\hat{v}'$ will be created by substituting $\mathcal{N}_i(v)$ with v in $\hat{v}^*$, and a combo-edge will be accordingly created by connecting $\hat{v}'$ to $\hat{v}^*$. As a result, after finding the combo-neighbors of $\hat{v}^*$ at hop $\ell = 1$, $\tilde{\mathcal{G}}$ becomes a star-network at center $\hat{v}^*$. We will then repeat the above procedures (i.e. star-sampling) for every newly found combo-node to find their combo-neighbors at hop $\ell + 1$ (which meanwhile also finds the edges among combo-nodes within the previous hop ℓ), until the subgraph size limit Q or the maximum hop $\ell_{\max}$ is reached.

By constructing the combo-graph and sampling subgraphs from it, we can efficiently traverse the combinatorial space by progressively moving around the combo-subgraph center while preserving diversified combinations of distant nodes under a local modeling approach, which will be discussed in the following section.

3.2 Graph Gaussian Processes Surrogate

After constructing the combo-subgraph, we can build a surrogate model for the expensive underlying function on this local region with graph Gaussian Processes ($\mathcal{GP}$). Specifically, we consider the normalized graph Laplacian $\tilde{\mathbf{L}}$: $\tilde{\mathbf{L}} = \mathbf{I} - \tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}$ for a combo-subgraph $\tilde{\mathcal{G}}$, where $\tilde{\mathbf{A}}$ is the adjacency matrix and $\tilde{\mathbf{D}}$ is the degree matrix. Then, the eigendecomposition of the graph Laplacian matrix is given by $\tilde{\mathbf{L}} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$, in which $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \cdots, \lambda_n)$ are the eigenvalues sorted in ascending order and $\mathbf{U} = [\boldsymbol{u}_1, \cdots, \boldsymbol{u}_n]$ are their corresponding eigenvectors. Now let $i, j \in \{1, \cdots, n\}$ be two indices of combo-nodes on $\tilde{\mathcal{G}}$, the covariance function (or *kernel*) $k(\hat{v}_i, \hat{v}_j)$ between an arbitrary combo-node pair $\hat{v}_i$ and $\hat{v}_j$ can be formulated in the form of a regularization function $r(\lambda_p)$ [46] defined on the eigenvalues $\{\lambda_p\}_{p=1}^n$:

Algorithm 1 Recursive combo-subgraph sampling

Input: Original (unknown) graph $\mathcal{G}$; The focal combo-node $\hat{v}^*$; Combo-subgraph size Q; Max neighbor hop $\ell_{\max}$.

Initialize: An combo-subgraph $\tilde{\mathcal{G}} = \{\tilde{\mathcal{V}}, \tilde{\mathcal{E}}\}$ with $\tilde{\mathcal{V}} = \{\hat{v}^*\}$ $\tilde{\mathcal{E}} = \varnothing$; Starting hop $\ell \leftarrow 1$; Set of newly found combo-nodes $\tilde{\mathcal{V}}_{new} \leftarrow \{\hat{v}^*\}$.

Define: $\texttt{Recursive\_Sampler}(\mathcal{G}, \tilde{\mathcal{G}}, \tilde{\mathcal{V}}_{new}, Q, \ell)$

1: **for** $\hat{v}$ in $\tilde{\mathcal{V}}_{new}$ **do**
2:     **for** v in $\hat{v}$ **do**
3:         Reveal the neighbors $\mathcal{N}(v)$ of v in $\mathcal{G}$.
4:         **for** v' in $\mathcal{N}(v) \cap \hat{v}$ **parallelly do**
5:             Generate a new combo-node by $\texttt{CONCAT}([v', \hat{v}\backslash v])$ and then create a combo-edge by connecting it to $\hat{v}$.
6:         **end for**
7:         Update $\{\tilde{\mathcal{V}}, \tilde{\mathcal{E}}\}$ in $\tilde{\mathcal{G}}$.
8:     **end for**
9:     **if** $|\tilde{\mathcal{V}}| > Q$ or $\ell > \ell_{\max}$ **then**
10:         Randomly drop the extra combo-nodes.
11:         **return** The combo-subgraph $\tilde{\mathcal{G}}$ of size Q.
12:     **end if**
13: **end for**
14: Update $\tilde{\mathcal{V}}_{new} \leftarrow \tilde{\mathcal{V}}\backslash\tilde{\mathcal{V}}_{new}$; $\ell \leftarrow \ell + 1$
15: **return** $\texttt{Recursive\_Sampler}(\mathcal{G}, \tilde{\mathcal{G}}, \tilde{\mathcal{V}}_{new}, Q, \ell)$

$$k(\hat{v}_i, \hat{v}_j) = \sum_{p=1}^n r^{-1}(\lambda_p)\boldsymbol{u}_p[i]\boldsymbol{u}_p[j], \tag{2}$$

where $\boldsymbol{u}_p[i]$ and $\boldsymbol{u}_p[j]$ are the i-th and j-th elements in the p-th eigenvector $\boldsymbol{u}_p$, and $r(\lambda_p)$ is some scalar-valued function for regularization. We refer readers to Appendix §E for discussion on a collection of commonly used kernels on graphs under the form of Equation (2).

3.3 Bayesian Optimization on the Combo-graph

With the structural combinatorial space and techniques to sample and build surrogate models on the combo-subgraphs, we now introduce the proposed GraphComBO framework in detail. For simplicity, we consider maximization in the following paragraphs, where the overall structure can be found in Figure 1 with key procedures summarized in Algorithm 2 and complexity discussed in Appendix §C.

Combo-subgraphs as trust regions. As discussed earlier, performing global modeling directly for combinatorial problems is usually impractical. Thus, inspired by the *trust region* method popularly used in continuous numerical optimization [8], reinforcement learning [43] and BO under other settings [19, 51], we take a local modeling approach on the combo-graph during the BO search. Starting with a random location (i.e. a combo-node $\hat{v}_0$) or a reasonable guess from domain knowledge, a combo-subgraph $\tilde{\mathcal{G}}_0$ will be constructed at center $\hat{v}_0$ by Algorithm 1. We will then move around

5

this combo-subgraph $\tilde{\mathcal{G}}_t$ at each iteration t on the combo-graph by changing its focal combo-node guided by the surrogate model and acquisition function, which will be explained shortly.

In particular, we introduce a hyperparameter Q that caps the combo-subgraph size to control the computational cost for the surrogate $\mathcal{GP}$, and then use the queried combo-node inside $\tilde{\mathcal{G}}_t$ as the training set $\mathcal{D}_t$ to fit the model (i.e. update the hyperparameters in its kernel). The acquisition function $\alpha(\hat{v})$ is then applied on the rest of unvisited combo-nodes in $\tilde{\mathcal{G}}_t$, and we select the combo-node $\hat{v}_t = \arg\max_{\hat{v}\in\tilde{\mathcal{V}}_t} \alpha(\hat{v})$ as the next location to query the underlying function. Here, any commonly used kernel and acquisition function are compatible with our setting, and we adopt the popular *diffusion* kernel [40] with *Expected Improvement* acquisition [26] in our experiments.

After querying the next location, we re-select the best-queried combo-node $\hat{v}_t^*$ in our training set $\mathcal{D}_t[\hat{v}]$ by choosing $\hat{v}_t^* = \arg\max_{\hat{v}\in\mathcal{D}_t[\hat{v}]} f(\hat{v})$, and compare it to the previous best location $\hat{v}_{t-1}^*$. If the best-queried value improves (i.e. $f(\hat{v}_t^*) > f(\hat{v}_{t-1}^*)$), the combo-subgraph in the next iteration $\tilde{\mathcal{G}}_{t+1}$ will be resampled at this new location $\hat{v}_t^*$ with Algorithm 1, or otherwise remains the same as $\tilde{\mathcal{G}}_t$. The search algorithm then continues until a querying budget T is reached, and we report the best-queried combo-node as the final result $\hat{v}_T^* = \arg\max_{\hat{v}\in\hat{v}_{1:T}} f(\hat{v})$.

Balancing exploration and exploitation. Similar to the continuous domain, the exploration-exploitation trade-off is also a fundamental concern when using BO on the proposed combo-graph, and we introduce two additional techniques to strike a balance between these two matters.

1. `failtol` that controls the tolerance of "failures" by counting continuous non-improvement steps. Once reached, the algorithm will restart at a new location using `restart_method`.

2. `restart_method` that either restarts at a random combo-node, the best-visited combo-node, or the initial starting location if specified.

Algorithm 2 BO for node-subsets on graphs

Input: Original (unknown) graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$; underlying function f defined on k- node subsets $\mathcal{S} \in \binom{|\mathcal{V}|}{k}$; combo-subgraph size Q; # failures threshold `failtol`; # queries T.

Objective: Find $\mathcal{S}_T^* = \arg\max_{\mathcal{S}_i\in\{\mathcal{S}_i\}_{i=1}^T} f(\mathcal{S}_i)$.

Initialize: Set initial training set $\mathcal{D}_0 \leftarrow \varnothing$ and queried set $\mathcal{O}_0 \leftarrow \varnothing$; Set restart status `restart` $\leftarrow$ `True`; Set counter of non-improvement tolerance $F \leftarrow 0$. Use initial `start_location` $\hat{v}_0$ if applicable, and specify the `restart_method` for `restart`.

1: **for** $t = 1, \cdots, T$ **do**
2:   **if** `restart` **then**
3:     Re-initialize the starting location $\hat{v}_t$ using `restart_method` and `start_location`; Query $\hat{v}_t$ and reset the training set $\mathcal{D}_t \leftarrow (\hat{v}_t, y_t)$; Set `restart` $\leftarrow$ `False`.
4:   **end if**
5:   Sample a combo-subgraph $\tilde{\mathcal{G}}_t = \{\tilde{\mathcal{V}}_t, \tilde{\mathcal{E}}_t\}$ with $|\tilde{\mathcal{V}}_t| = Q$ centered at the **best** training combo-node $\hat{v}_{t-1}^*$ from $\mathcal{D}_{t-1}[\hat{v}]$ using Algorithm 1 in §3.1.
6:   Fit the $\mathcal{GP}$ surrogate defined in §3.2 on $\tilde{\mathcal{G}}_t$ by maximum likelihood, with the queried combo-nodes inside $\tilde{\mathcal{G}}_t$ being the training set (i.e. $\mathcal{D}_t[\hat{v}] = \tilde{\mathcal{V}}_t \cap \mathcal{O}_{t-1}[\hat{v}]$).
7:   Optimize the acquisition function α; Select $\hat{v}_t = \arg\max_{\hat{v}\in\tilde{\mathcal{V}}_t} \alpha(\hat{v})$ from $\tilde{\mathcal{G}}_t$ as the next query; Obtain the function value y_t at $\hat{v}_t$.
8:   Update query set $\mathcal{O}_t \leftarrow \mathcal{O}_{t-1} \cup (\hat{v}_t, y_t)$ and training set $\mathcal{D}_t \leftarrow \mathcal{D}_{t-1} \cup (\hat{v}_t, y_t)$; Select the best training combo-node $\hat{v}_t^* = \arg\max_{\hat{v}\in\mathcal{D}_t[\hat{v}]} f(\hat{v})$.
9:   Update $F \leftarrow F + 1$ **if** $\hat{v}_t^* = \hat{v}_{t-1}^*$; Set `restart` $\leftarrow$ `True` **if** `failtol` $= F$ **or** all combo-nodes in $\tilde{\mathcal{G}}_t$ are queried.
10: **end for**
11: **return** $\hat{v}_T^* = \arg\max_{\hat{v}\in\mathcal{O}_T[\hat{v}]} f(\hat{v})$.

In addition, the combo-subgraph size Q, which can be viewed as the "volume" of the trust region under graph setting, also controls the step size of exploration. These strategies together can cohesively assist GraphComBO in adapting to various tasks. For example, a small `failtol` will encourage exploration in the combinatorial space when `restart_method` is set to a random combo-node, which is useful when optimizing an underlying function with low graph signal smoothness [15]. By contrast, when increasing `failtol` and setting `restart_method` to the best-queried combo-node, the algorithm will exploit more around the local optimal and is hence more suitable for smoother functions. §K further provides an ablation study on these hyperparameters.

Impact of the underlying function f and the subset size k. It is natural to expect that the interaction between the underlying function f and graph structure, which relates to signal smoothness over the combo-graph, will exert a significant influence on the search performance. Specifically, the

6

Figure 3: Results for synthetic problems on BA, WS, SBM and 2D-Grid networks with $k = [4, 8, 16, 32]$, where `Regret` indicates the difference between ground truth and the best query so far.

optimization is expected to be challenging either when f is less correlated to the graph structure even if the latter is informative (e.g. random noise on a BA network [4] as an extreme case), or when f is correlated to the graph structure but the latter is non-informative (e.g. eigenvector centrality on a Bernoulli random graph [18]). In the meantime, as the subset size k increases, exploration will become more expensive when using a combo-subgraph of fixed size Q or fixed number of hops ℓ_{max}. Recall that in **Lemma 3.2** where we state that at most ℓ elements will be changed between two combo-nodes that are ℓ-hops away, it implies that more queries are required to exhaust all possible modifications of the elements in the subset when its size k increases. Empirical findings from our experiments in §4 further corroborate these hypotheses, where we also provide detailed discussions on model behavior in §G and kernel performance under different levels of signal smoothness in §F.

Relation to previous BO methods with graph settings. While BO has been combined with graph-related settings to find the optimal *graph structures* such as in the literature of NAS [27, 40, 42] and graph adversarial attacks [50], it remains largely under-explored for optimizing functions defined on the *nodes* or *node subsets* in the graph. Although one recent work `BayesOptG` [52] considered such novel setups, it only considered functions defined on a single node, which can be viewed as a special case in our setting when $k = 1$. The construction of the "combo-graph" in our approach shares similarity with the construction of the combinatorial graph in `COMBO` [40]; however, the problems being addressed there do not arise in a natural graph setting, and we present a more detailed discussion of the related work in Appendix §A, together with an additional experiment for comparison in §H.

4 Experiments

Setups. We conduct comprehensive experiments on four synthetic problems and five real-world tasks to validate our proposed framework, where readers are also referred to the appendix for discussions on: §B detailed experimental settings with task descriptions and visualizations; §E validation of common kernels on graphs under our settings; §G a thorough analysis of GraphComBO's underlying behavior; and §K ablation studies on the hyper-parameters. We closely follow the standard setups in BO literature [3, 19, 23]. Specifically, we query 300 times and repeat 20 times with different

7

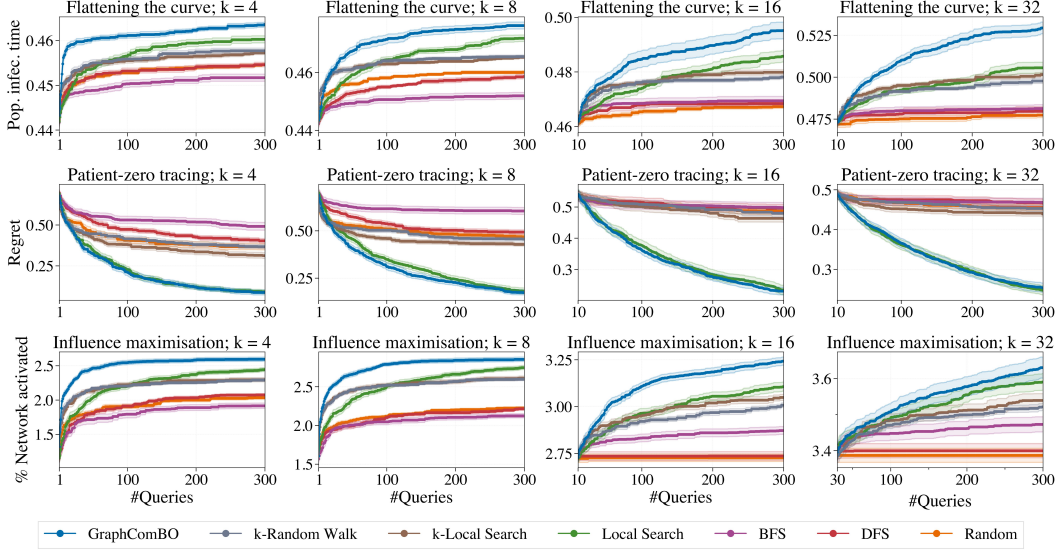



Figure 4: Results for flattening the curve, patient-zero tracing and influence maximization.

random seeds for each task, in which the mean and standard error of the cumulative optima are reported for all methods. For simplicity, we use a diffusion kernel [40] with automatic relevance determination and adopt *Expected Improvement* [26] as the acquisition function to investigate subset sizes of $k = [4, 8, 16, 32]$, where we also fix $Q = 4,000$ and `failtol` $= 30$ across all experiments. In addition, we also initialize the algorithm with 10 queries using simple random walks when $k \geqslant 16$.

Baselines. As the proposed framework is a first-of-its-kind BO method for optimizing expensive and black-box functions of node subsets on generic graphs, we consider three graph-traversing algorithms that operate on the original graph: *Random*, *k-Random Walk* and *k-Local Search*; and three algorithms on the proposed combo-graph: *BFS*, *DFS* and *Local Search* as the baseline methods, with their details described in Appendix §B. Notably, the local search method, which randomly queries a neighbor of the best-queried combo-node at each iteration, can be viewed as a BO method that uses a "random" surrogate model, and hence serves as a good indicator for GraphComBO's behavior.

Synthetic problems on random graphs. We first validate the proposed framework on four ubiquitous random graph types with commonly used analytical underlying functions. Concretely, we consider Barabási-Albert (BA) [4], Watts-Strogatz (WS) [54], stochastic block model (SBM) [22] and 2D-grid networks, where their corresponding "base" underlying functions are eigenvector centrality, degree centrality, PageRank [6], and Ackley function [1], respectively. We then take the average over node values inside a subset to obtain the final underlying function, which, given the analytical setting, enables us to compute the difference (`Regret`) between the queried-best value and ground truth. The search results are presented in Figure 3, where we also explained the problem settings in detail in Appendix §B.1 and summarized the graph statistics in Table 1.

Real-world optimization tasks. After validation under synthetic settings, we carry out five real-world experiments on epidemic contact networks, social networks, transportation networks, and molecule graphs, where their results are presented in Figure 4 and Figure 5. The statistics of the underlying functions and graphs are summarized in Table 1, where the detailed setting for each scenario is explained and visualized in §B.2-B.6. Specifically, we consider the following tasks:

- *Flattening the curve in epidemics (§B.2).* We adopt the widely-used SIR simulations [29] on a real-world contact network with a goal of protecting k nodes in the network, such that the expected time of reaching 50% population infection will be maximally delayed.

- *Identifying patient-zero in communities (§B.3).* We apply SIR on an SBM network to simulate a disease contagion across multiple communities, where the goal is to identify k individuals with the earliest infection time, given the complete transmission network not known a priori.

8

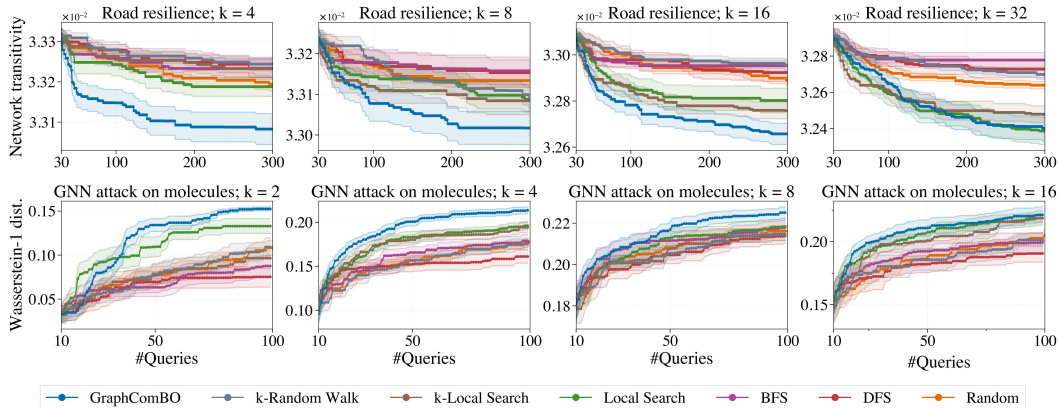



Figure 5: Results for road resilience testing and GNN attacks on molecules with edge-masking.

- ***Maximizing influence on social networks (§B.4).*** We consider the influence maximization problem on a social network [45] with independent cascading simulations [28], in which we aim to select the optimal k nodes as the seeds (i.e. source of influence) that maximize the expected number of final influenced individuals (represented as a fraction of the network size).

- ***Resilience testing on transportation networks (§B.5).*** The objective of this task is to identify the k most vulnerable roads (edges), such that their removal will lead to the maximal drop in a certain utility function measuring the operation status (estimated by network transitivity).

- ***Black-box attacks on graph neural networks (§B.6).*** Considering a graph-level GNN pre-trained for molecule classification [39] with a particular input graph, we conduct a challenging black-box attack with no access to the model parameter but only a limited number of queries for its output. Our goal here is to mask k edges such that the output from the victim GNN (at softmax) will be maximally perturbed from the original output, as measured by the Wasserstein distance.

Discussion on results. We can observe that the proposed GraphComBO framework generally outperforms all the other baselines with a clear advantage on both synthetic and real-world tasks. It is worth noting that such gain in performance seems to be diminishing as k increases and, in certain scenarios, BO also tends to perform similarly to local search. While these phenomena are generally consistent with our previous hypothesis in §3.3, we further provide the following explanations to attain a better understanding of the model's underlying behavior.

1. Given a combo-subgraph with a fixed size Q, we tend to capture structural information in a smaller neighborhood due to the increase of combo-node degrees. First, when k increases, the combo-node degree will increase linearly (as discussed in **Lemma 3.3**). Second, if the synthetic underlying function has a strong positive correlation with node degree, such as eigenvector centrality, the degree of the center combo-node will increase as the search progresses. Both factors will lead to a smaller neighborhood around the focal node (in terms of shortest path distance) covered by the combo-subgraph, which in turn means more steps are required to explore beyond the current region, especially when the algorithm reaches a local optimum.

2. As discussed in §3.3, underlying functions with low signal smoothness will negatively affect BO's performance, which partially explains the comparable results between BO and local search on WS and SBM where the graph structures are less informative, as well as on the molecule network where the underlying function involves a graph neural network and is relatively non-smooth compared to other tasks. In addition, as the kernels used in $\mathcal{GP}$ (§3.2) come with an underlying assumption on function smoothness, the surrogate model will capture less signal information when fitting a less-smooth function, thus making BO behave similarly to a random model (i.e. the local search). To better support this claim, we further conduct a sensitivity analysis of the kernels to signal smoothness at different levels in Appendix §F.

Further analysis on model behaviors. Readers are also referred to Appendix §G for a more detailed behavior analysis that elaborates on the above explanations and Appendix §K for a thorough ablation study on Q and `failtol`. In addition, Appendix §H provides a comparison with

9

COMBO [40] on small-scale networks, Appendix §I tests our framework on a large social network OGB-arXiv with $|\mathcal{V}| = 1.7 \times 10^5$, and finally Appendix §J discusses the framework's performance under a noisy setting where observations are corrupted at different noise levels.

5   Conclusion and Future Works

In this work, we introduce a novel Bayesian optimization framework to optimize black-boxed functions defined on node subsets in a generic and potentially unknown graph. By constructing a tailored combinatorial graph and sampling subgraphs progressively with a recursive algorithm, we are able to traverse the combinatorial space and optimize the objective function using BO in a sample-efficient manner. Results on both synthetic and real-world experiments validate the effectiveness of the proposed framework, and we use detailed analysis to study its underlying behavior.

On the other hand, we have also identified the following limitations during our experiments, which can be explored as future directions for this line of work.

- As discussed in the paper, the performance of BO gradually deteriorates when the subset size k increases. In this sense, some modifications are expected to better control the combinatorial explosion while preserving useful information from the underlying graph structure.
- The proposed framework adopts a local modeling approach inspired by the trust region method to control the computational cost. However, we expect some improvement in BO's performance if we inject some global information (if available) into surrogate modeling, such as using some self-supervised method with a graph neural network to replace the Laplacian embedding.
- The current algorithm adopts a fixed strategy for hyperparameters like subgraph size Q and maximum hop $\ell_{\max}$, where we believe the optimization would benefit from a more flexible design such as a self-adaptive Q and $\ell_{\max}$ as the search continues.
- In all experiments, we assume no prior knowledge of the problem and adopt a random initialization method before the search. However, it is also an important direction to explore when a good starting location or certain characteristics of the function are available from domain knowledge.

We believe the proposed combo-graph would bring new insights to a broader community of machine learning research on graphs. While there are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

Acknowledgment

H.L. is funded by the ESRC Grand Union Doctoral Training Partnership and the Oxford-Man Institute of Quantitative Finance. X.D. acknowledges support from the EPSRC (EP/T023333/1). The authors thank members of the Machine Learning Research Group and Oxford-Man Institute of Quantitative Finance for discussing initial ideas and experiments.

References

[1] Ackley, D. *A connectionist machine for genetic hillclimbing*, volume 28. Springer science & business media, 2012.

[2] Arora, A., Galhotra, S., and Ranu, S. Debunking the myths of influence maximization: An in-depth benchmarking study. In *Proceedings of the 2017 ACM international conference on management of data*, pp. 651–666, 2017.

[3] Baptista, R. and Poloczek, M. Bayesian optimization of combinatorial structures. In *International Conference on Machine Learning*, pp. 462–471. PMLR, 2018.

[4] Barabási, A.-L. and Albert, R. Emergence of scaling in random networks. *science*, 286(5439): 509–512, 1999.

[5] Boeing, G. Osmnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks. *Computers, environment and urban systems*, 65:126–139, 2017.

10

[6] Brin, S. and Page, L. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1-7):107–117, 1998.

[7] Chen, W., Lin, T., Tan, Z., Zhao, M., and Zhou, X. Robust influence maximization. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 795–804, 2016.

[8] Conn, A. R., Gould, N. I., and Toint, P. L. *Trust region methods*. SIAM, 2000.

[9] Cui, J., Tan, Q., Zhang, C., and Yang, B. A novel framework of graph bayesian optimization and its applications to real-world network analysis. *Expert Systems with Applications*, 170: 114524, 2021.

[10] De Arruda, G. F., Barbieri, A. L., Rodríguez, P. M., Rodrigues, F. A., Moreno, Y., and da Fontoura Costa, L. Role of centrality for the identification of influential spreaders in complex networks. *Physical Review E*, 90(3):032812, 2014.

[11] de Ocáriz Borde, H. S., Arroyo, A., Morales, I., Posner, I., and Dong, X. Neural latent geometry search: Product manifold inference via gromov-hausdorff-informed bayesian optimization. *Advances in Neural Information Processing Systems*, 2023.

[12] Defferrard, M., Bresson, X., and Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in Neural Information Processing Systems*, 29: 3844–3852, 2016.

[13] Deshwal, A. and Doppa, J. Combining latent space and structured kernels for bayesian optimization over combinatorial spaces. *Advances in neural information processing systems*, 34:8185–8200, 2021.

[14] Deshwal, A., Belakaria, S., and Doppa, J. R. Mercer features for efficient combinatorial bayesian optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 7210–7218, 2021.

[15] Dong, X., Thanou, D., Frossard, P., and Vandergheynst, P. Learning laplacian matrix in smooth graph signal representations. *IEEE Transactions on Signal Processing*, 64(23):6160–6173, 2016.

[16] Dong, X., Thanou, D., Toni, L., Bronstein, M., and Frossard, P. Graph signal processing for machine learning: A review and new perspectives. *IEEE Signal processing magazine*, 37(6): 117–127, 2020.

[17] Engsig, M., Tejedor, A., Moreno, Y., Foufoula-Georgiou, E., and Kasmi, C. Domirank centrality reveals structural fragility of complex networks via node dominance. *Nature Communications*, 15(1):56, 2024.

[18] ERDOS, P. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci.*, 5:17–60, 1959.

[19] Eriksson, D., Pearce, M., Gardner, J., Turner, R. D., and Poloczek, M. Scalable global optimization via local bayesian optimization. *Advances in neural information processing systems*, 32, 2019.

[20] Garnett, R. *Bayesian optimization*. Cambridge University Press, 2023.

[21] Haklay, M. and Weber, P. Openstreetmap: User-generated street maps. *IEEE Pervasive computing*, 7(4):12–18, 2008.

[22] Holland, P. W., Laskey, K. B., and Leinhardt, S. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

[23] Hvarfner, C., Stoll, D., Souza, A., Lindauer, M., Hutter, F., and Nardi, L. πbo: Augmenting acquisition functions with user beliefs for bayesian optimization. In *Tenth International Conference of Learning Representations, ICLR 2022*, 2022.

11

[24] Imani, M. and Ghoreishi, S. F. Graph-based bayesian optimization for large-scale objective-based experimental design. *IEEE transactions on neural networks and learning systems*, 33 (10):5913–5925, 2021.

[25] Jiang, J., Wen, S., Yu, S., Xiang, Y., and Zhou, W. Identifying propagation sources in networks: State-of-the-art and comparative studies. *IEEE Communications Surveys & Tutorials*, 19(1): 465–481, 2016.

[26] Jones, D. R., Schonlau, M., and Welch, W. J. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13:455–492, 1998.

[27] Kandasamy, K., Neiswanger, W., Schneider, J., Poczos, B., and Xing, E. P. Neural architecture search with bayesian optimisation and optimal transport. *Advances in neural information processing systems*, 31, 2018.

[28] Kempe, D., Kleinberg, J., and Tardos, É. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 137–146, 2003.

[29] Kermack, W. O. and McKendrick, A. G. A contribution to the mathematical theory of epidemics. *Proceedings of the royal society of london. Series A, Containing papers of a mathematical and physical character*, 115(772):700–721, 1927.

[30] Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2014.

[31] Korovina, K., Xu, S., Kandasamy, K., Neiswanger, W., Poczos, B., Schneider, J., and Xing, E. Chembo: Bayesian optimization of small organic molecules with synthesizable recommendations. In *International Conference on Artificial Intelligence and Statistics*, pp. 3393–3403. PMLR, 2020.

[32] Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J., and Glance, N. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 420–429, 2007.

[33] Li, Y., Fan, J., Wang, Y., and Tan, K.-L. Influence maximization on social graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 30(10):1852–1872, 2018.

[34] Liu, W. and Song, Z. Review of studies on the resilience of urban critical infrastructure networks. *Reliability Engineering & System Safety*, 193:106617, 2020.

[35] Maier, B. F. and Brockmann, D. Effective containment explains subexponential growth in recent confirmed covid-19 cases in china. *Science*, 368(6492):742–746, 2020.

[36] McKay, R. A. *Patient Zero and the Making of the AIDS Epidemic*. University of Chicago Press, 2017.

[37] Mockus, J. The application of bayesian methods for seeking the extremum. *Towards global optimization*, 2:117, 1998.

[38] Mockus, J. and Mockus, J. *The Bayesian approach to local optimization*. Springer, 1989.

[39] Morris, C., Kriege, N. M., Bause, F., Kersting, K., Mutzel, P., and Neumann, M. Tudataset: A collection of benchmark datasets for learning with graphs. *arXiv preprint arXiv:2007.08663*, 2020.

[40] Oh, C., Tomczak, J., Gavves, E., and Welling, M. Combinatorial bayesian optimization using the graph cartesian product. *Advances in Neural Information Processing Systems*, 32, 2019.

[41] Ru, B., Alvi, A., Nguyen, V., Osborne, M. A., and Roberts, S. Bayesian optimisation over multiple continuous and categorical inputs. In *International Conference on Machine Learning*, pp. 8276–8285. PMLR, 2020.

12

[42] Ru, B., Wan, X., Dong, X., and Osborne, M. Interpretable neural architecture search via bayesian optimisation with weisfeiler-lehman kernels. *International Conference on Learning Representations*, 2021.

[43] Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015.

[44] Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., and De Freitas, N. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2015.

[45] Shchur, O., Mumme, M., Bojchevski, A., and Günnemann, S. Pitfalls of graph neural network evaluation. *Relational Representation Learning Workshop NeurIPS*, 2018.

[46] Smola, A. J. and Kondor, R. Kernels and regularization on graphs. In *Learning Theory and Kernel Machines: 16th Annual Conference on Learning Theory and 7th Kernel Workshop, COLT/Kernel 2003, Washington, DC, USA, August 24-27, 2003. Proceedings*, pp. 144–158. Springer, 2003.

[47] Stehlé, J., Voirin, N., Barrat, A., Cattuto, C., Isella, L., Pinton, J.-F., Quaggiotto, M., Van den Broeck, W., Régis, C., Lina, B., et al. High-resolution measurements of face-to-face contact patterns in a primary school. *PloS one*, 6(8):e23176, 2011.

[48] Sun, J. and Tang, J. A survey of models and algorithms for social influence analysis. *Social network data analytics*, pp. 177–214, 2011.

[49] Sun, L., Dou, Y., Yang, C., Zhang, K., Wang, J., Philip, S. Y., He, L., and Li, B. Adversarial attack and defense on graph data: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[50] Wan, X., Kenlay, H., Ru, R., Blaas, A., Osborne, M. A., and Dong, X. Adversarial attacks on graph classifiers via bayesian optimisation. *Advances in Neural Information Processing Systems*, 34:6983–6996, 2021.

[51] Wan, X., Nguyen, V., Ha, H., Ru, B., Lu, C., and Osborne, M. A. Think global and act local: Bayesian optimisation over high-dimensional categorical and mixed search spaces. In *International Conference on Machine Learning*, pp. 10663–10674. PMLR, 2021.

[52] Wan, X., Osselin, P., Kenlay, H., Ru, B., Osborne, M. A., and Dong, X. Bayesian optimisation of functions on graphs. *Advances in Neural Information Processing Systems*, 2023.

[53] Wang, X., Jin, Y., Schmitt, S., and Olhofer, M. Recent advances in bayesian optimization. *ACM Computing Surveys*, 55(13s):1–36, 2023.

[54] Watts, D. J. and Strogatz, S. H. Collective dynamics of 'small-world'networks. *nature*, 393 (6684):440–442, 1998.

[55] Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

[56] Zhang, Z.-K., Liu, C., Zhan, X.-X., Lu, X., Zhang, C.-X., and Zhang, Y.-C. Dynamics of information diffusion and its applications on complex networks. *Physics Reports*, 651:1–34, 2016.

13

A   Related Work

BO for combinatorial optimization.   Bayesian optimization has been widely applied to solving combinatorial problems with black-box and expensive-to-evaluate underlying functions [44, 53]. Notably, `BOCS` [3] handles the combinatorial explosion of the discrete search space by utilizing an approximate optimizer for the acquisition function, which addresses the limited scalability of common acquisition functions to large combinatorial domains; `CoCaBo` [41] further tackles the setting of mixed search space with multiple categorical variables by introducing a $\mathcal{GP}$ kernel to capture the interaction between continuous and categorical inputs; and `LADDER` [13] takes a latent variable approach that first encodes the problem into a latent space via unsupervised learning and then adopts a structure-coupled kernel, which integrates both decoded structures and the latent representation for better surrogate modeling. While BO has provided a sample-efficient way for the above combinatorial optimization problems, its extension to settings with graph structures, especially when the search space itself is a generic graph, still remains largely under-explored and will be discussed in the following section.

BO with graphs.   Despite several works in the literature combining BO with graph-related settings, the majority of them focus on optimization over graph inputs (i.e. each configuration itself is a graph, and the goal is to optimize for graph structures). In particular, `NASBOT` [27] treats the neural network architecture as a graph structure and uses BO to perform neural architecture search, while `NAS-BOWL` [42] approaches the same problem from a different perspective by using Weisfeiler-Lehman kernels with BO. Other examples include using BO for molecular graph designs [31], graph adversarial attack [50], and a general framework for optimizing functions on graph structures [9]. These works, however, are under a different setup compared to our current work, which aims to optimize functions defined on node subsets in a single generic and potentially unknown graph. Moreover, the above works typically seek for a global vector-embedding of the graph configuration, after which standard kernels will be applied to measure their similarity in the Euclidean space.

On the other hand, `BayesOptG` [52] proposes a framework that employs BO to optimize functions defined on a single node on graphs, where the search space is a generic graph and the configurations are nodes on the graph. The similarity between two configurations is then measured between node pairs with kernels on graphs capturing structural information. As mentioned earlier, our paper is a generalization of this previous work by considering a combinatorial setting, in which `BayesOptG` can be viewed as a special case under our framework when the number of nodes in the subset is $k = 1$.

Lastly, another relevant line of works from the literature is `COMBO` [40] and its variants [24, 14, 11], where the underlying function is defined on a Cartesian product graph computed from k small graphs. In particular, each node on this combinatorial graph represents a combination of k elements from the k graphs, after which kernels on graphs are leveraged to measure the similarity between nodes in the combinatorial space. Nevertheless, the combinatorial graph introduced in our work differentiates substantially from that in `COMBO`, and we emphasize the differences in the following.

1. From the problem setting, `COMBO` is designed for k-node combinations from k distinct graphs $\{G_i\}_{i=1}^k$ (i.e. one node from each graph, which corresponds to one variable in the combinatorial optimization). The structure of these graphs and the resulting combinatorial graph is therefore pre-defined and fully available. In contrast, our work is concerned with the combination of k nodes (a k-node subset) from a single and generic graph whose structure (and hence the structure of our combo-graph) is potentially unknown a priori.

2. The resulting search space for `COMBO` is of dimension $\prod_{i=1}^k N_i$ with N_i corresponding to the size of ith graph G_i, whereas in our work, the combinatorial space has a dimension $\binom{N}{k}$ with N being the size of G. Note that in `COMBO`, even if we have k identical graphs and naively calculate their Cartesian product, the resulting space N^k is still not the same as our case. For instance, (a, b) and (b, a) are two distinct sets in `COMBO`, but they should be considered as one single set in our scenario. Besides, sets with duplicated elements (e.g., (a, a) or (a, a, b)) are valid in `COMBO` but meaningless in our setting, which will introduce redundant computational costs.

3. One of the main contributions of `COMBO` is that the eigendecomposition of the large combinatorial graph can be performed on the k smaller graphs with Kronecker product operation. Nevertheless, it is limited to small k and N_i as the memory is simply not large enough to fit in an eigenbasis of size $(\prod_{i=1}^k N_i)^2$ even with moderate choices of k and N_i.

14

B Experimental Details

Experimental setups. This section provides the details of our synthetic and real-world experimental settings, where we summarize the statistics of the underlying graphs and functions in Table 1, and then visualize them in Figure 6 for synthetic problems and Figure 7 for real-world problems.

Table 1: Summary statistics of the underlying graphs used in our experiments.

Type	Underlying Function	Search Space	Underlying Graph	# Nodes	# Edges
Synthetic	Avg. Eigenvector centrality	Node subsets	BA ($m = 5$)	$10k$	$50k$
	Avg. Degree centrality	Node subsets	WS ($k = 10, p = 0.1$)	$1k$	$5k$
	Avg. PageRank scores	Node subsets	SBM (# clusters $= 4$)	$1k$	$7k$
	Avg. Ackley function	Node subsets	2D-GRID	$5k$	$10k$
Real-world	Population infection time	Node subsets	Proximity contact network	223	$6k$
	Individual infection time	Node subsets	SBM (# clusters $= 4$)	$1k$	$7k$
	Expected # nodes influenced	Node subsets	Coauthor network (CS)	$18k$	$163k$
	Network transitivity	Edge subsets	Manhattan road networks	$5k$	$8k$
	Change of GNN predictions	Edge subsets	Molecule graphs (ENZYMES)	37	84

We closely follow the standard setups in BO literature [3, 19, 23] and investigate subset sizes of $k = [4, 8, 16, 32]$, where we use $Q = 4000$, `failtol` $= 30$ and set `restart_method` to the best-queried combo-node for all experiments. Specifically, we query 300 times and repeat 20 times with different random seeds for each task, in which the mean and standard error of the cumulative optima are reported for all methods. For simplicity, we use a diffusion kernel [40] with automatic relevance determination and adopt *Expected Improvement* [26] as the acquisition function.

Similar to standard BO setups, we initialize our algorithm with 10 queries by simple random walks on the original graph when $k \geqslant 16$, except for the influence maximization experiment where we use 30 initial queries at $k = 32$ since the underlying graph is relatively large with $|\mathcal{V}| \approx 18k$. In addition, we also use simple random initialization of 30 queries for Ackley on 2D-grid and road resilience testing experiments; and 10 queries for GNN attack on molecules, since their underlying graphs contain relatively weak structural information, which are not suitable for graph-related initialization methods such as random walk. After evaluating the initial query locations, we select the best query as the starting point for all baselines to ensure a fair comparison.

Hardware and running time. All the experiments are conducted on a computing cluster of 96 `Intel-Xeon@2.30GHz` CPU cores with 250 GB working memory. The running times for synthetic problems are typically under 1 hour with parallel computing. For real-world problems, except for the simulation-based experiments that take around 12 hours due to the evaluation of large numbers of simulations, the rest experiments will normally be finished within 1 hour with parallel computing.

Baselines. We consider the following baselines in our experiments.

- **Random Search** which randomly samples k nodes from the original graph at each iteration and performs well compared to other graph-based methods if the underlying function is less smooth on the combinatorial graph or less correlated with the original graph structure.

- **k-Random Walk** which maintains k independent random walks on the original graph that forms a subset of k nodes at each step, featuring a fast-exploration characteristic from the starting nodes, and works particularly well for exploration-heavy tasks.

- **k-Local Search** takes a similar approach with the `k-Random Walk` baseline on the original graph. However, it will only proceed to the next neighbors if the current k-node subset is a better query compared to the previous ones, otherwise, it will hold at the same nodes and re-execute the random walk until a better location is found.

- **BFS and DFS** which explore function values on graphs by starting with an initial node and then traveling the graph according to depth or breadth. Specifically, BFS exploits all nodes at the current depth before moving to the next depth, while DFS explores nodes as far as possible along each branch before backtracking. Both methods operate on the combo-graph.

15

- **Local Search** which also travels on the the proposed combo-graph by randomly selecting a node from the neighbors of the best-visited node at each iteration. If all neighbors of the best node have been queried, the algorithm will then restart from new a random location. Notably, this method can also be viewed as a BO using a random surrogate with no acquisition function and hence serves as a good indicator for BO's behavior.

B.1 Synthetic Experiments

In this section, we introduce the random graphs and synthetic functions used in our synthetic experiments, where a visualization can be found in Figure 6.

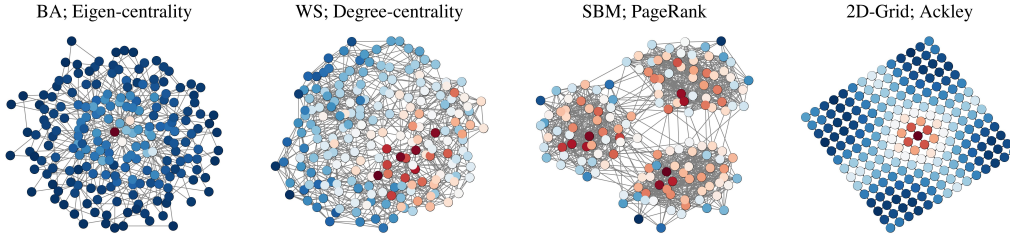



Figure 6: Visualization of random graphs and underlying functions used in our synthetic experiments. Specifically, node color represents *eigenvector centrality* on BA network, *degree centrality* on WS network, *PageRank* on SBM network, and *Ackley* function on 2D-Grid. Note that under the synthetic settings, we take the **average** over the k elements within a node subset as the underlying function.

Random graphs used in experiments. We consider the Barabási-Albert (BA) [4] network, Watts-Strogatz (WS) [54] network, stochastic block model (SBM) [22], and 2D-grid as the underlying graphs in our synthetic experiments, which are explained in the following part.

- **Barabási-Albert network** is constructed using a preferential attachment mechanism. Specifically, we start with m_0 initial nodes and then gradually add new nodes one at a time, where each new node v_i is connected to m existing nodes with a probability $P(v_i)$ proportional to the number of links that the existing nodes already have, which can be mathematically expressed as:

$$P(k_i) = \frac{k_i}{\sum_j k_j},$$

  where k_i is the degree of node v_i and the sum is over all pre-existing nodes.

- **Watts-Strogatz network** explains the "small-world" phenomena in a variety of networks by interpolating between a regular lattice and a random graph. In particular, the model first starts with a regular ring lattice of N nodes where each node is connected to K nearest neighbors (i.e. $NK/2$ total edges), then rewires $K/2$ edges for each node with probability $p \in [0, 1]$ to a random node in the network while avoiding self-loops and duplicate edges.

- **Stochastic Block Model** divides N nodes into K communities where each community i has a predetermined size N_i. Then, the probability of an edge between nodes in cluster i and j is defined by a matrix $\mathbf{P}$ of size $K \times K$, where $\mathbf{P}_{ij}$ represents the probability of an edge between nodes in cluster i and cluster j. The adjacency matrix $\mathbf{A}$ of the network is then generated from $\mathbf{P}$ such that entry $\mathbf{A}_{uv}$ for an arbitrary node-pair u and v is a Bernoulli random variable:

$$\mathbf{A}_{uv} \sim \text{Bernoulli}(\mathbf{P}_{c_u c_v}),$$

  where c_u and c_v are the cluster memberships of nodes u and v, respectively. In our experiment, for simplicity, we consider an SBM of $1k$ nodes with $K = 4$ clusters in equal size, and fix the inter-cluster probability $p_{in} = 5 \times 10^{-2}$ and intra-cluster probability $p_{out} = 10^{-3}$.

Synthetic underlying functions used in experiments. We consider eigenvector centrality, degree centrality, PageRank scores, and Ackley function as the (base) underlying function on the aforementioned random graphs. Note that we first use these base functions to assign a scalar value to each node in the graph, and then take the average within the subset as the final underlying function. Such

16

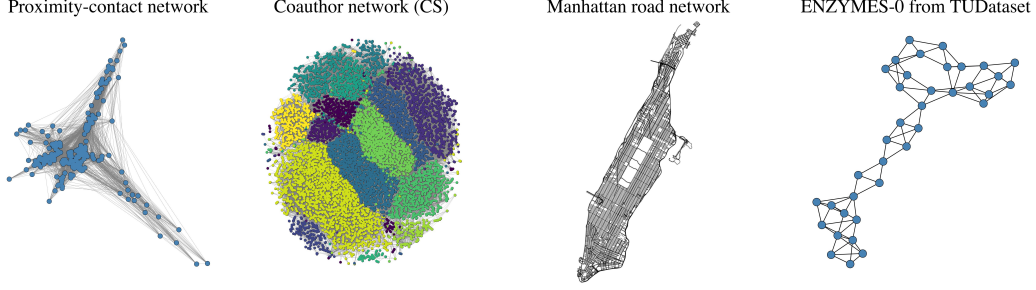



Figure 7: Visualization of the real-world networks used in our experiments.

synthetic setup will enable us to track the difference between our current best query and the ground truth, which is denoted as `Regret` (to minimize) when we present the results.

- **Eigenvector centrality** is a measure of the influence of a particular node v_i in the network, which is defined as the solution $\mathbf{x}$ to the following equation:

$$\mathbf{Ax} = \lambda\mathbf{x},$$

  where λ is the largest eigenvalue of the adjacency matrix $\mathbf{A}$. Note that the solution $\mathbf{x}$ is also the eigenvector corresponding to λ, and hence the name eigenvector centrality.

- **Degree centrality** measures the number of links incident to a particular node v_i, which, for an undirected network of N nodes, can be expressed as:

$$DC(v_i) = \frac{\deg(v_i)}{N-1},$$

  where $\deg(v_i)$ is the degree of node v_i. The underlying intuition is to normalize the degree of each node by the maximum possible degree $N-1$ in the network.

- **PageRank** is a variant of eigenvector centrality originally developed by Google [6] to rank web pages according to their "importance" scores. It incorporates random walks with a damping factor and can be mathematically formulated in the following form:

$$PR(v_i) = \frac{1-d}{N} + d \sum_{v_j \in \mathcal{N}(v_i)} \frac{PR(v_j)}{\deg^{\text{out}}(v_j)},$$

  where d is the damping factor (typically set to 0.85), N is the total number of nodes, $\mathcal{N}(v_i)$ denotes the in-neighbors of node v_i, and $\deg^{\text{out}}(v_j)$ is the out-degree of node v_j.

- **Ackley function** is a non-convex function with multiple local optima and has been widely used for testing optimization algorithms, which can be expressed in the following 2-D form:

$$f(x,y) = -20\exp\left(-0.2\sqrt{0.5\left(x^2 + y^2\right)}\right) - \exp(-0.5(\cos 2\pi x + \cos 2\pi y)) + 20 + \exp(1).$$

  Additionally, a random Gaussian noise $\sigma = 0.5$ is added to the original function to alter the smoothness property of the graph signal defined over the 2D-grid:

$$\hat{f}(x,y) = f(x,y) + \epsilon, \text{ with } \epsilon \sim N(0,\sigma^2).$$

  Note that as the underlying graph (2D-grid) in this experiment contains little structural information compared to other graphs such as BA and WS, instead of using the random walk initialization, we adopt a simple random initialization method of 30 queries before searching and use the best query as the starting location for all methods.

B.2 Flattening the curve in Epidemics

In this experiment, our goal is to protect an optimal subset of k nodes (individuals) in a contact network to maximally slow down an epidemic process simulated by a diffusion model SIR [29]. The contact network [47] used in this experiment is collected by proximity sensors from a primary school in France, which contains 236 nodes and $5,899$ edges.

17

The *Suspicious*, *Infected*, *Recovered* simulation model.   In the SIR model based on a network $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, each node has three statuses: *Suspicious*, *Infected* and *Recovered*. Starting with a fraction of p initial infectious nodes in the population, at time step $t \in \{1, ..., T\}$, each infected node has a probability β to infect another node if they are neighbors, and meanwhile has a probability γ to transit to Recovered status. Once a node is at Recovered status, it can not be infected again (e.g. quarantined, immune, or vaccinated). More formally, let $\mathbf{x}_{v,t} \in \{I, S, R\}$ denote the node status (Infected, Susceptible, Recovered) and $\mathcal{S}_{I,t}, \mathcal{S}_{S,t}, \mathcal{S}_{R,t}$ denote the set of nodes in each category at time t, the model can be mathematically formulated as:

$$\forall v \in \mathcal{S}_{I,t}, \begin{cases} \mathbb{P}\left[\mathbf{x}_{v,t+1} = R\right] = \gamma \\ \mathbb{P}\left[\mathbf{x}_{v,t+1} = I\right] = 1 - \gamma \end{cases} \tag{3}$$

$$\forall v \in \mathcal{S}_{S,t}, \begin{cases} \mathbb{P}\left[\mathbf{x}_{v,t+1} = I\right] = 1 - (1 - \epsilon) \times (1 - \beta)^{|N(v) \cap \mathcal{S}_{I,t}|} \\ \mathbb{P}\left[\mathbf{x}_{v,t+1} = S\right] = (1 - \epsilon) \times (1 - \beta)^{|N(v) \cap \mathcal{S}_{I,t}|} \end{cases} \tag{4}$$

$$\forall v \in \mathcal{S}_{R,t}, \mathbb{P}\left[\mathbf{x}_{v,t+1} = R\right] = 1 \tag{5}$$

where $\epsilon \in [0, 1]$ is a parameter representing a probability of spontaneous infection from unknown factors [52]. Since the above model implies a random simulation process, a large number of Monte Carlo samples is typically required when estimating the expectation of certain functions based on SIR, which is often expensive to evaluate and optimize.

Flattening the curve with SIR.   As healthcare resource is often limited (e.g. hospitals, vaccines, quarantine centers), one is usually interested in slowing down the transmission speed of the epidemic process by protecting the most "important" individuals, which prevents the public health system from breaking down due to the sudden shortage of its capacity [35]. We demonstrate this idea with SIR in the above contact network with $N = 100$ simulations in Figure 8, in which each run has an initial fraction of $p = 0.1$ population infected, an infection rate of $\beta = 10^{-3}$, a recovery rate of $\gamma = 10^{-2}$, and no spontaneous infection $\epsilon = 0$. Note that the same settings have been used in the main experiments.

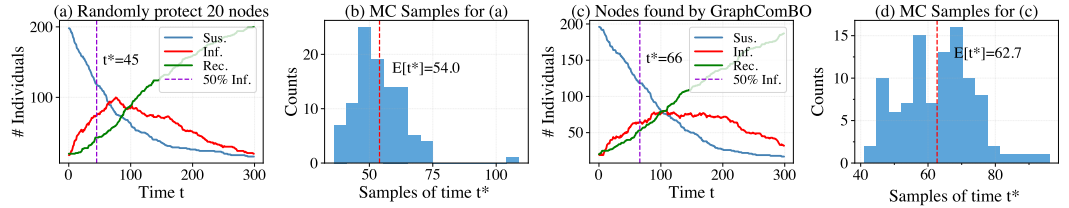



Figure 8: Demonstration of SIR and its simulations on the real-world proximity contact network.

To protect nodes from infection, we set the chosen k-node subset to the Recovered status at the beginning of each simulation, then record the time t^* that 50% population is infected. After obtaining the results from $N = 100$ simulations, we record the mean infection time $\mathbb{E}[t^*]$ as our underlying function, which we aim to maximize (i.e. delay the time when reaching half-population infection). From Figure 8, we can observe a clear curve-flattening effect when protecting 20 nodes selected by BO (plots c & d) compared to randomly choosing 20 nodes (plots a & b) in the network. Note that to make the surrogate fitting more numerically stable, we also map $\mathbb{E}[t^*]$ to the range $[0, 1]$ by dividing a constant of the maximal number of iteration $T = 120$.

B.3   Tracing Patient-zero in the Community

In disease transmission analysis such as AIDS and COVID-19, one would be interested in identifying the earliest individuals (patient-zero) infected in the community [36, 52]. Nevertheless, such a process is often time-consuming since it requires interviewing patients to obtain their infection dates and then gradually revealing the transmission network by interviewing their close contacts (i.e., the graph is not known a prior), as illustrated in Figure 9.

18

explored nodes · currently revealed neighbours · ← target node to query next

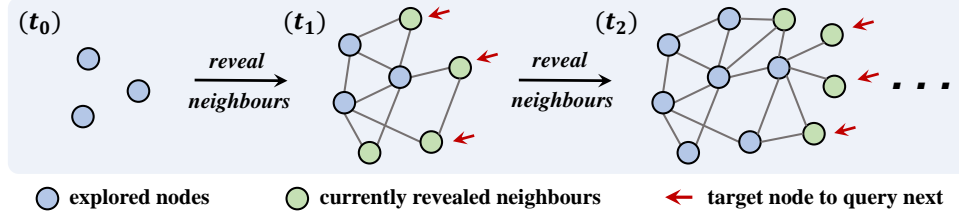



Figure 9: Demonstration of the patient-zero tracing setting, where the underlying contact network is not fully observed initially. To gradually reveal the graph structure, at each step t, we query k nodes (i.e. record the first times they are infected) and reveal their neighbors (e.g. interview the patients and obtain their contacts). The objective is to find the k patients of the earliest infection time.

Individual infection time. In this task, we use the aforementioned SIR model and SBM network to simulate a disease contagion across multiple communities, where the goal is to identify the earliest k individuals infected in the whole network. Specifically, at a certain time step T during the epidemic (set to $T = 100$ in the experiment), for every node in $\mathcal{S}_{I,T} \cup \mathcal{S}_{R,T}$ we denote τ_v as the time of infection for node v and map it into a scalar value $f(v) \in [0, 1]$ via the following transformation:

$$\forall v \in \mathcal{V}, f(v) = \begin{cases} 0 & \text{if } v \in \mathcal{S}_{S,T} \\ \left(1 - \frac{\tau_v}{T}\right)^2 & \text{if } v \in \mathcal{S}_{I,T} \cup \mathcal{S}_{R,T} \end{cases} \tag{6}$$

Then, we take the average within the k-node subset as the final underlying function in this experiment, which is maximized when the subset corresponds to the earliest k nodes of infection.

Note that in this experiment only a single SIR simulation is required, and we run 20 times with different random seeds to report the results. For reference, the parameters used in each run are: initial fraction of infected population $p = 0.5\%$, infection rate & recovery rate of $\beta = \gamma = 10^{-2}$, spontaneous infection rate of $\epsilon = 0.5\%$, and simulation time step of $T = 100$.

B.4 Influence Maximization on Social Networks

The *Influence Maximization* (IM) problem over social networks has been widely applied to marketing and recommendations [33], where the goal is to select the optimal k nodes as the seeds (i.e. source of influence) that maximize the expected number of final influenced individuals, which is typically estimated by *Independent Cascading* (IC) simulation [28] and its variants. In this experiment, we consider the vanilla IC simulations on an academic collaboration network (Coauthor CS [45]) of $18,333$ nodes and $163,788$ edges, with detailed settings discussed in the following.

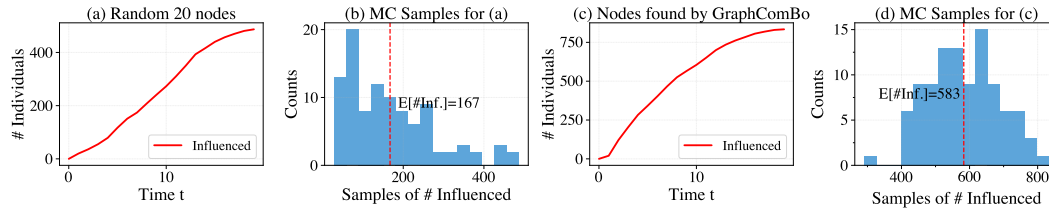



Figure 10: Demonstration of Independent cascading with simulations on the CS coauthor network.

Independent cascading. The IC model takes a similar setup to the SIR model above, except that each node only takes one of the two statuses: *Activated* and *Inactivated*. Starting with an initial subset of k activated nodes, at each time step $t \in \{1, ..., T\}$, the activated nodes will have a one-shot probability of p to activate their neighbors. Once a node is activated, it will remain activated until the end of the process when there is no more new node to be activated. Likewise, calculating the expectation of functions based on IC also requires a large number of Monte Carlo simulations, which is hence computationally expensive in most cases.

Influence maximization with IC. The task of IM is to select the optimal set of k nodes as the seeds of influence, such that the expected number of activated nodes at the end of IC is maximized.

19

In our experiment, we set the activation rate to $p = 0.05$ and used $N = 1000$ Monte Carlo samples to estimate the expected number of final influenced individuals. In particular, the underlying function is set to be $\#\text{Activated}/|\mathcal{V}|$, that is, the fraction of activated nodes in the network. Figure 10 demonstrates the simulation results from two different strategies of selecting $k = 20$ initial seeds: random selection (plots a & b) and GraphComBO (plots c & d), and we can observe a clear difference in their expected numbers of final activated nodes.

B.5 Road Resilience Testing on Transportation Networks

The resilience of an infrastructure network denotes its ability to maintain normal operations when facing certain disruptions [34], and one is often interested in identifying and protecting the most important nodes or edges to avoid catastrophic failure. In this experiment, we investigate the resilience of Manhattan road networks in New York City using `OSMnx` [5], a Python tool built on `OpenStreetMap` [21]. Figure 7 provides a visualization of the network, where each node denotes an intersection and each edge represents a road (with network_type set to "drive")

The objective here is to identify the k most vulnerable roads, such that their removal will lead to the maximal drop in a certain utility function, which could be difficult to evaluate in practice if it involves simulations or real-world queries. For experimental purposes, we use network transitivity (global clustering coefficient) as a proxy estimation for this utility function (to minimize), which measures the global connectivity of the graph based on the number of triangles:

$$\text{Transitivity}(\mathcal{G}) = \frac{\#\text{Triangles}}{\#\text{Triads}}, \tag{7}$$

where `Triad` means two edges with a shared vertex. Similar to the Ackley on 2D-grid experiment, as the underlying graph is a grid-like road network, we utilize simple random initialization of 30 queries before searching and use the best query as the starting location for all baselines.

Line-graph for functions of edge subsets. Since the underlying function here is defined on edge (road) subsets, we will first change the original graph $\mathcal{G}$ into its line graph $\mathcal{G}_{line}$, where each node on the line graph $\mathcal{G}_{line}$ represents an edge in $\mathcal{G}$, and two nodes on $\mathcal{G}_{line}$ are adjacent if and only if their corresponding edges in $\mathcal{G}$ share a common endpoint. Then, the line graph will be used as the underlying graph in our proposed framework. Note that this procedure is independent of the underlying function evaluation, which is still on the original graph with certain black-box processes.

B.6 Black-box Attacks on Graph Neural Networks

In this task, we conduct adversarial attacks on graph neural networks (GNN) under a challenging black-box setting [49], where the attacker has no access to model parameters but only a limited number of queries for the outputs. Considering a GNN pre-trained for graph-level classification tasks, for a particular target graph under attack, our goal is to mask k edges on this input graph, such that the output from GNN (at softmax) will be maximally perturbed from the original output.

Perturbation via edge-masking. Concretely, we use GIN [55] as the victim GNN and pre-train it on the TUDataset ENZYMES [39] for small molecule classification, and measure the change in GNN prediction after perturbation by the Wasserstein distance, which can be formulated as follows:

$$f(\mathcal{S}) = W_1\Big(g(\mathcal{G}),\; g\big(\Phi(\mathcal{G}, \mathcal{S})\big)\Big), \tag{8}$$

where $f(\mathcal{S})$ is the underlying function defined on the k-edge subset $\mathcal{S}$ (to maximize), W_1 is the empirical Wasserstein-1 distance, g denotes the pre-trained GNN at softmax, and $\Phi(\mathcal{G}, \mathcal{S})$ is a perturbation on $\mathcal{G}$ by masking the k-edge subset $\mathcal{S}$, which will lead to a perturbed graph $\mathcal{G}'$.

For reference, the GIN model has 3 hidden layers with 64 hidden dimension and is trained for 200 epochs by Adam [30] with a learning rate of 10^{-3}. The pre-trained model achieves 99.5% accuracy on the dataset, and we use the first graph (index=0) from the dataset as the target graph under attack. Note that no training/testing split is needed here as we are conducting attacks on pre-trained GNN.

Since the underlying function is defined on edge subsets of small molecule graphs, we will use the above line graph operation again and only search for 100 queries, where simple random initialization is also adopted for 10 queries before searching.

20

C   Algorithm Complexity

Overall, the computational complexity of the proposed method for optimizing a function defined on k nodes in a graph of size N mainly comes from (1) constructing a combo-subgraph of size Q and (2) fitting the surrogate model.

1. Suppose at recursion $\ell \in \{1, 2, 3, ...\}$ we found M_ℓ new combo-nodes ($M_0 = 1$). We will first need $kM_{\ell-1}$ dictionary lookup operations to find the neighbors of nodes in the k-node subset for each $M_{\ell-1}$ input combo-nodes, and then M_ℓ concatenations to create M_ℓ new combo-nodes. The recursion repeats until $\sum_{\ell=0}^{L} M_\ell \geqslant Q$ at $\ell = L$, which leads to a total of $k \sum_{\ell=0}^{L-1} M_\ell \leqslant kQ$ dictionary lookups at $\mathcal{O}(kQ)$, plus around Q concatenations at $\mathcal{O}(Q)$ (the last recursion L will break before finish when we reach Q).

2. The computational cost consists of two parts: (a) the Graph Fourier Transformation (GFT) and (b) computing the predictive posterior in Gaussian Processes (GP) using the Gaussian conditioning rule.

   (a) The GFT requires eigendecomposing the graph Laplacian matrix: $\mathbf{L} = \mathbf{U\Lambda U}^\top$, which typically costs $\mathcal{O}(N^3)$ for a graph of N nodes.
   (b) To obtain the predictive posterior from a GP by Gaussian conditioning rules (explained in section 2 line 105), we need to compute the inverse of the kernel matrix $K_{1:t}^{-1}$ for the observed t datapoints, which requires $\mathcal{O}(t^3)$. Since $t <= N$, the maximum complexity for this term is also at $\mathcal{O}(N^3)$ for a graph of N nodes.

However, since the surrogate model operates on a subgraph of size Q, we can limit the computational cost to $\mathcal{O}(Q^3)$ and only need to re-construct the combo-subgraph and re-compute its eigenbasis when the center changes (i.e. when finding a better query location). Note that the computational cost from (a) is at $\mathcal{O}(Qk)$ which is insignificant compared to $\mathcal{O}(Q^3)$ in practice. This also leads to an efficient memory consumption where we only need to store a combo-subgraph of size Q with its cached eigenbasis (a $Q \times Q$ matrix) during the search.

D   Proofs of Lemmas in §3.1

In this section, we provide proofs for **Lemma 3.2** and **Lemma 3.3** in §3.1.

Lemma 3.2. *In the proposed combo-graph, at most ℓ elements in the subset will be changed between any two combo-nodes that are ℓ-hop away.*

Proof. Considering an arbitrary combo-node $\hat{v}_i = (v_i^{(1)}, v_i^{(2)}, ..., v_i^{(k)})$ of k elements as the center of an ℓ-hop ego combo-subgraph on the proposed combinatorial graph $\tilde{\mathcal{G}}^{<k>}$. According to **Definition 3.1**, there will be strictly 1 element in difference between two neighboring combo-nodes, which implies that the 1st-hop neighbors of $\hat{v}_i$ will have one different element, the 2nd-hop neighbors will have one or two different element(s), the 3rd-hop will have one, two, or three different element(s), ..., and by induction, we can conclude that at hop-ℓ there will be at most ℓ elements in difference. $\square$

Lemma 3.3. *The degree of combo-node $\hat{v}_i$ increases linearly with k and is maximized by the subset of nodes with top k degrees:* $\deg(\hat{v}_i) = \sum_{j=1}^{k} |\mathcal{N}(v_i^{(j)}) \backslash \{v_i^{(j')}\}_{j' \neq j}^k|$.

Proof. The degree $\deg(\hat{v}_i)$ of an arbitrary combo-node $\hat{v}_i = (v_i^{(1)}, v_i^{(2)}, ..., v_i^{(k)})$ is a linear combination over k constant terms, where each term $j \in \{1, ..., k\}$ equals the number of neighbors $\mathcal{N}(v_i^{(j)})$ of an element node $v_i^{(j)}$ that are not inside the combination. As the maximum term is capped by $|\mathcal{V}| - 1$, which is the largest possible degree in the original graph of an arbitrary structure, we conclude that the combo-node degree will increase linearly with k. $\square$

E   Details of the Kernels on Graphs

Common choices of kernels on graphs.   Following the discussion in §3.2, we analyze the performance of four kernels on the combinatorial graph and their details are summarized in Table 2.

21

Table 2: Summary of some common kernels on graphs.

Choice of Kernel	Regularization $r(\lambda_p)$	Kernel $K_n(\mathcal{V}, \mathcal{V})$
Diffusion	$\exp(\beta_p \lambda_p)$	$\sum_{p=1}^{n} \exp(-\beta_p \lambda_p) \boldsymbol{u}_p \boldsymbol{u}_p^\top$
Polynomial	$\sum_{j=1}^{\eta-1} \beta_j \lambda_p^j + \epsilon$	$\sum_{p=1}^{n} \left(\sum_{j=1}^{\eta-1} \beta_j \lambda_p^j + \epsilon \right)^{-1} \boldsymbol{u}_p \boldsymbol{u}_p^\top$
Sum-of-Inverse Polynomials	$\left(\sum_{j=1}^{\eta-1} (\beta_j \lambda_p^j + \epsilon)^{-1} \right)^{-1}$	$\sum_{p=1}^{n} \left(\sum_{j=1}^{\eta-1} (\beta_j \lambda_p^j + \epsilon)^{-1} \right) \boldsymbol{u}_p \boldsymbol{u}_p^\top$

Specifically, we consider: **Polynomial** [12] that consists of polynomials of the eigenvalue at order $\eta \in \mathbb{Z}_{\geqslant 1}$, where the hyperparameters are the coefficient for each order; **Sum-of-Inverse Polynomials** [52] which is a variant of the polynomial kernel that takes a scaled harmonic mean of different degrees; **Diffusion** [40] that penalizes the magnitude of the frequency (eigenvalue), and we also consider its implementation with the automatic relevance determination (ARD) strategy.

The polynomial and sum-of-inverse polynomials kernels have η hyperparameters $\boldsymbol{\beta} = [\beta_0, \cdots, \beta_{\eta-1}]^\top$ that are constrained to be non-negative to ensure a positive semi-definite covariance matrix. Meanwhile, we maintain the settings in the previous work [52] that set η to be $\min\{5, \texttt{diameter}\}$, which strikes a balance between expressiveness and regularisation. Whereas in the diffusion kernel, there are n hyperparameters $\boldsymbol{\beta} = [\beta_1, \cdots, \beta_n]$ to be learned and are sometimes prone to over-fitting when n is large.

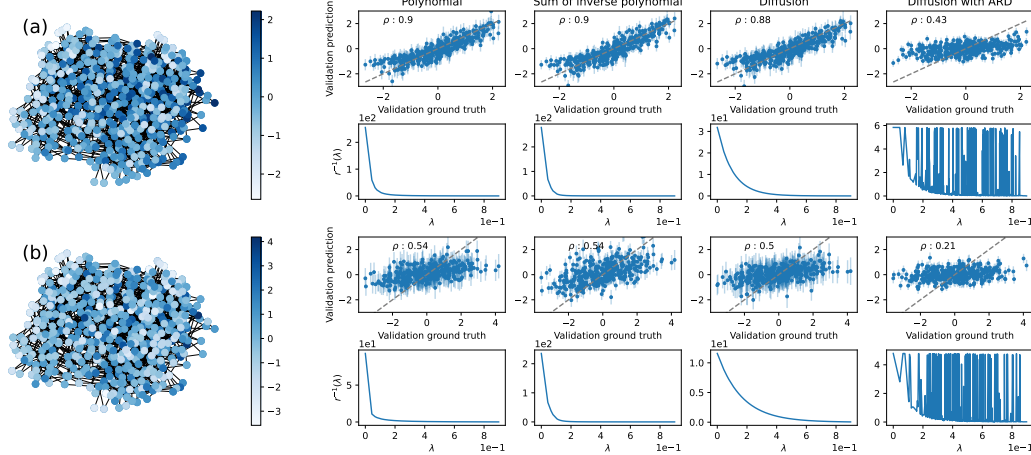



Figure 11: Kernel validation on the combinatorial graph based on a BA network ($n = 20, m = 2$). To design the underlying function, we take the elements from the third eigenvector and average them over $k = 3$ nodes. Specifically, (a) shows the results on the testing data measured by Spearman's correlation coefficient ρ, and (b) shows the results when adding Gaussian noise to the ground truth.

Kernel validation. We consider a synthetic setting on two 20-node networks with subset size $k = 3$: a BA network ($m = 2$) and a WS network $(k, p) = (5, 0.2)$, where the combinatorial graph in both networks contains $\binom{20}{3} = 1140$ combo-nodes. The underlying function is designed in the following way: we first perform eigen-decomposition on the graph Laplacian matrix: $\tilde{\mathbf{L}} = \mathbf{U} \boldsymbol{\Lambda} \mathbf{U}^\top$ where $\tilde{\mathbf{L}} = \mathbf{I} - \tilde{\mathbf{D}}^{-1/2} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-1/2}$ is the normalised graph Laplacian. The eigenvalues $\boldsymbol{\Lambda} = \text{diag}(\lambda_1, \cdots, \lambda_n)$ are then sorted in ascending order and possess frequency information in the spectral domain [16], where smaller eigenvalues indicate lower frequencies. As such, their corresponding eigenvectors $\mathbf{U} = [\boldsymbol{u}_1, \cdots, \boldsymbol{u}_n]$ can be used as signals of different smoothness and will be discussed in more detail in Appendix §F. For the current experiment, we will take the elements from the eigenvector that corresponds to the 2nd non-zero eigenvalue (which is a smooth signal on the underlying graph), and use the average over k nodes as the underlying function in the combinatorial space.

After standardization, we use 25% combo-nodes as the training set to fit the models and validate their performance on the rest 75% combo-nodes with Spearman's rank-based correlation coefficient ρ. In addition, we also consider a noisy scenario where a Gaussian noise of $\sigma = 1$ is added to the original function, where their results are summarized in Figure 11 for BA and Figure 12 for WS. We

22

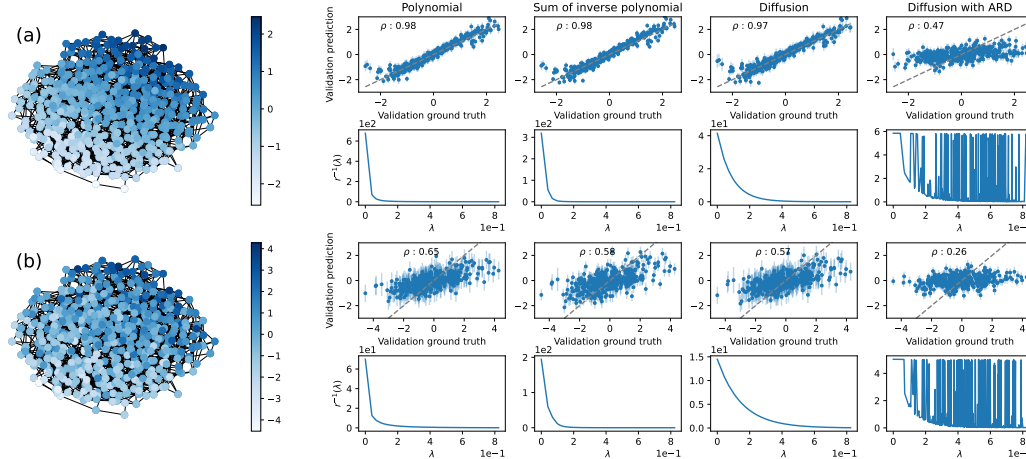



Figure 12: Kernel validation on the combo-graph based on a WS network ($n = 20, k = 5, p = 0.2$).

observe that all kernels can capture the original signal except for Diffusion with ARD, which learns a non-smooth transformation on the spectrum due to its over-parameterization. Nevertheless, we found the difference in performance is insignificant when using less-smooth underlying functions in §F.

F   Kernel Performance under Different Signal Smoothness

Following the setups in Appendix §E above, we now investigate how the inherited smoothness of the underlying function influences the performance of our kernels.

The smoothness of graph signals in the combinatorial space.   The graph Fourier transform is given by $\hat{f}(\Lambda) = \mathbf{U}^\top f$, which transforms the original graph signal f to the frequency domain, as illustrated in the second plot from Figure 13. To change the smoothness of the underlying function in the combinatorial space, we consider j-th eigenvector with $j \in [2, 4, 8, 12, 16]$ as the underlying signals (from the original graph) and then use the same method in §E that takes the average over the nodes in the subset as the underlying function in the combinatorial space. To compare the smoothness among different underlying functions (in the combinatorial space), we first calculate the cumulative energy of Fourier coefficients; since we are modeling on random graphs, the process will be repeated 50 times with different random seeds, after which we plot the results on the third plot in Figure 13 with mean and standard error. We can observe a clear trend that the underlying function in the combinatorial space becomes less smooth when using an eigenvector that corresponds to a higher frequency (larger eigenvalue).

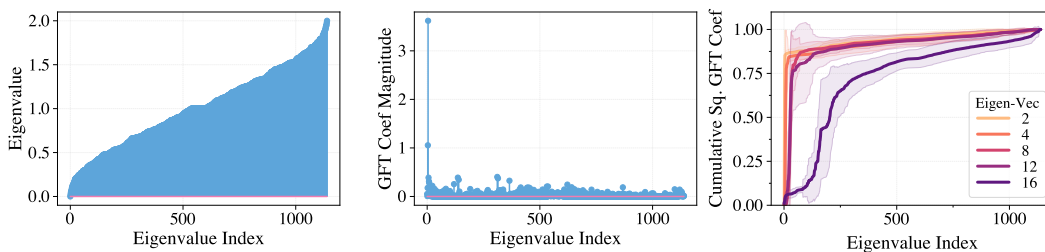



Figure 13: Smoothness of different underlying functions (average of different eigenvectors).

Kernel performance under different smoothness.   With the same settings in Appendix §E, we validate the performance of kernels on functions of different smoothness levels in the combinatorial graph (as described above) and report their results by Spearman's correlation coefficient ρ as a box-plot in Figure 14. For each kernel, we can see a clear drop in its validation performance as the function becomes less smooth, which will in turn negatively affect the performance of BO.

23

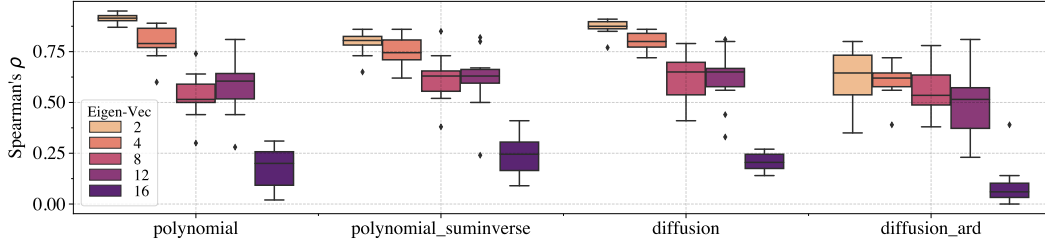



Figure 14: Performance (Spearman's rank-based correlation coefficient ρ) of kernels for underlying functions with different smoothness, where darker shades use eigenvectors of the higher index and thus indicate less-smooth functions.

G   Behavior Analysis of GraphComBO

In this section, we provide an in-depth behavior analysis of GraphComBO from two of the main experiments: a synthetic task of maximizing the average eigenvector centrality on BA networks, and a real-world task of flattening the curve on the contact network. The results are present in Figure 15 and Figure 16 respectively, where we also record additional information on (1) the explored combo-graph size and (2) the distance of the current combo-subgraph center to the starting location. Note that these recorders are only available for methods based on the proposed combo-graph, where all of these methods start at the same location before searching.

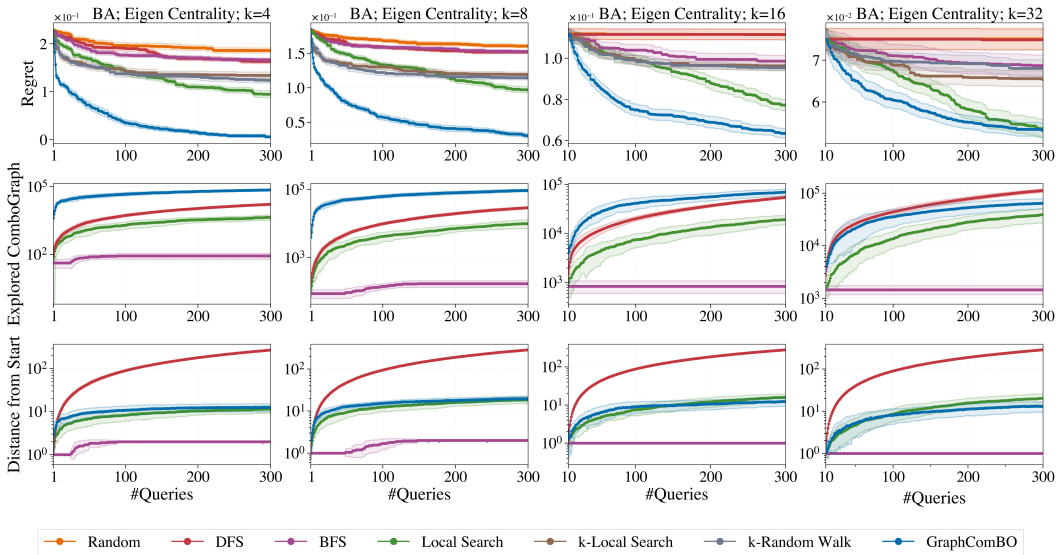



Figure 15: Behavior analysis of maximizing average eigenvector centrality on the BA network.

From both figures, it is straightforward to find that BFS and DFS behave differently given their exploration size and travel distance, in which BFS performs heavy exploitation and DFS performs large exploration. On the other hand, while BO explores more combinatorial space than local search in both experiments, we can notice the following distinctions in the source of its performance gain.

Considering the synthetic experiment results on BA network with a small k, we can observe that the subgraph center of GraphComBO is slightly more distant from the start location compared to the local search at the beginning, but later saturates and is caught up by local search. Such behavior also holds when k increases, especially at $k = 32$ where local search generally travels more distantly than GraphComBO. This implies that when k is small, the performance gain may mainly come from the exploration, whereas when k increases, we are losing the relative advantage of exploration and the performance gain is mainly from exploitation, which is consistent with our conjecture in §3.3. On the contrary, the result from flattening the curve experiment tells a different story, where GraphComBO takes a more exploitation-focused strategy compared to local search when k is small, but as k increases, it gradually shifts to an exploration-driven behavior.

24

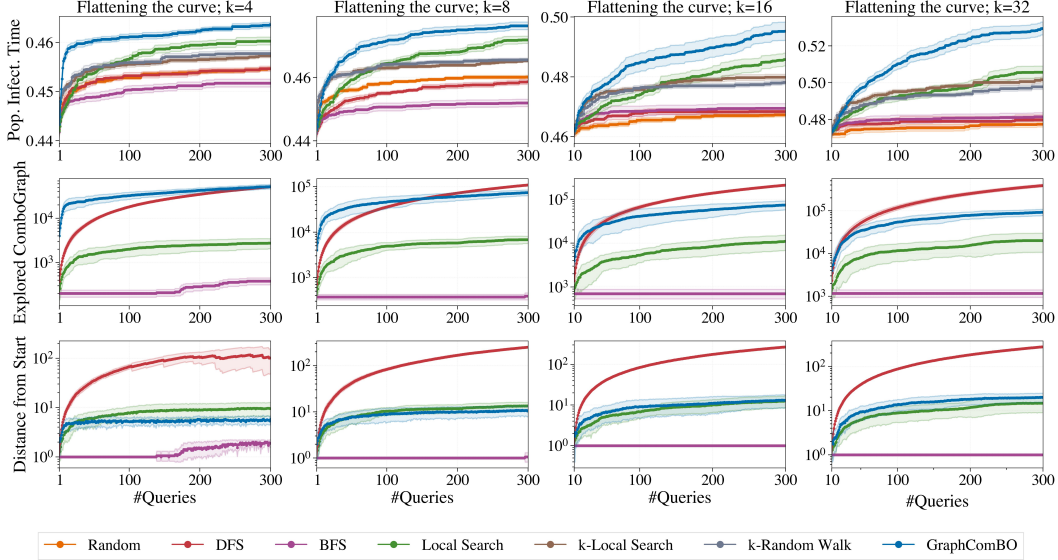



Figure 16: Behavior analysis of flattening the curve experiment on the contact network with SIR.

H Comparison with COMBO

In this section, we compare our method with COMBO [40] on small BA and WS graphs of $|\mathcal{V}| = 500$ (still much larger than the graphs used in COMBO's experiments), where the results in Figure 17 show a clear advantage of our framework over COMBO, and we make the following explanations.

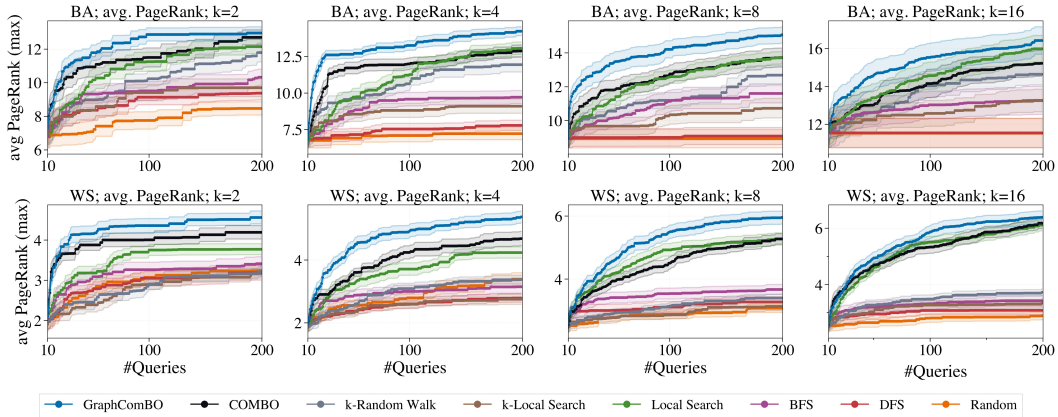



Figure 17: Comparison with COMBO in maximizing avg. PageRank on small BA and WS networks.

To implement COMBO under our setting of k-node subsets from a single graph $\mathcal{G}$ of size N, we generate k identical copies of $\mathcal{G}$ and form the k-node subset by drawing one node from each of the copy. This leads to a search space of N^k, which is the key limitation of COMBO under this setting since it is supposed to be $\binom{N}{k}$. As a result, there are many repeated and invalid locations in the search space, for example, at $k = 3$, $(1, 2, 3), (1, 3, 2), (2, 1, 3), ...$ are different subsets in COMBO, but they all should be the same subset under the current single graph setting; meanwhile $(1, 2, 2), (1, 1, 2), (1, 2, 1), ...$ are valid subsets in COMBO, but they are invalid k-node combinations on a single graph. This limitation makes COMBO highly inefficient under this new problem setting, and therefore leads to inferior performance compared to our proposed method.

25

I Scalability on Large Graphs

Results on OGB-arXiv. Since our framework assumes no prior knowledge of the full graph and takes a local modeling approach that gradually reveals the graph structure, it can scale to large underlying graphs with a reasonable choice of k. To better support this claim, we further test GraphComBO on a large social network OGB-arXiv ($|\mathcal{V}| = 1.7 \times 10^5$) from the open graph benchmark with k up to 128, where the results in Figure 18 show a clear advantage of our framework over the other baselines. Note that the local search methods underperform the random baseline under this setting, since exploration is relatively more important than exploitation.

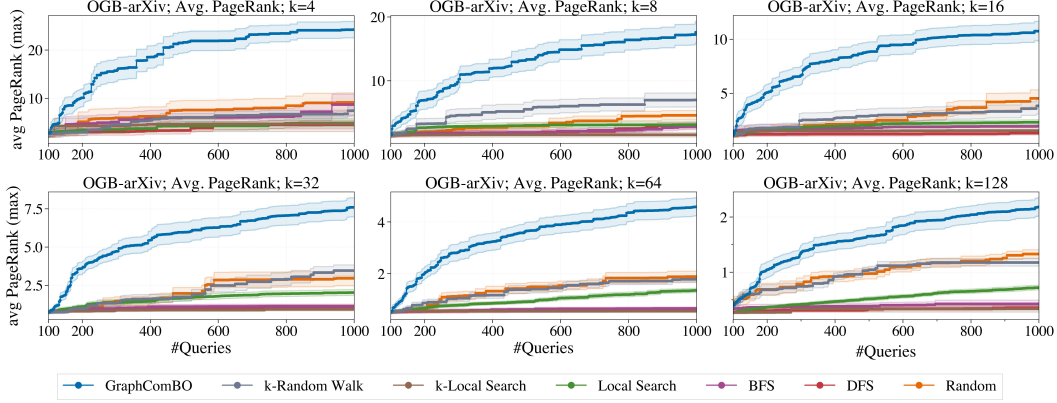



Figure 18: Maximizing avg. PageRank on the OGB-arXiv network ($|\mathcal{V}| = 1.7 \times 10^5$, $|\mathcal{E}| = 10^6$).

Choice of k in the experiments. The subset size k is set to $[2, 4, 8, 16, 32]$ across the experiments, which is a common paradigm in the literature of subset selection on graphs [28, 32, 7] with $k < 50$ (<1% of the network), and the problem has been proven to be NP-hard in many problems due to the combinatorial explosion in search space $\binom{N}{k}$, e.g. $\binom{1000}{32} \approx 2.3 \times 10^{60}$. As such, the diminishing performance gain w.r.t. subset size k poses a general challenge in the literature, and it becomes even more challenging in our setting, since the underlying function is fully black-boxed and we assume no prior information of the graph structure.

Nevertheless, the proposed method still generally outperforms the other baselines across all experiments, and in the least favorable case, it performs comparably to the local search, which is also a novel baseline introduced in our paper since it needs to operate on the proposed combo-graph.

J Settings under Noisy Observations

To show our framework's capability of handling noise, we further conduct a noisy experiment at different noise levels on BA ($|\mathcal{V}| = 10k$) and WS ($|\mathcal{V}| = 1k$) networks with $k = 8$, where the goal is to maximize the average PageRank within a node subset, i.e., $f(\mathcal{S}) = \frac{1}{k} \sum_{i=1}^{k} PageRank(\mathcal{S}_i)$. with $\mathcal{S}$ being a subset of k nodes $\{v_1, v_2, \dots, v_k\}$ in the underlying graph $\mathcal{G}$.

A standardized underlying function with noise. While it is difficult to show the noise level to the signal variance in real-world experiments because of the combinatorial space, we can construct a standardized signal under this synthetic setting with the following procedures. First, we standardized the PageRank scores over all nodes to mean=0 and std=1 in the original space (denoted as $PageRanks$). To standardize the underlying function in the combinatorial space, we multiply $\sqrt{k}$ to the average $PageRanks$ as the final underlying function $\tilde{f}(S)$, which is defined as follows:

$$\tilde{f}(\mathcal{S}) = \sqrt{k} f_s(\mathcal{S}) = \frac{1}{\sqrt{k}} \sum_{i=1}^{k} PageRanks(S_i), \tag{9}$$

26

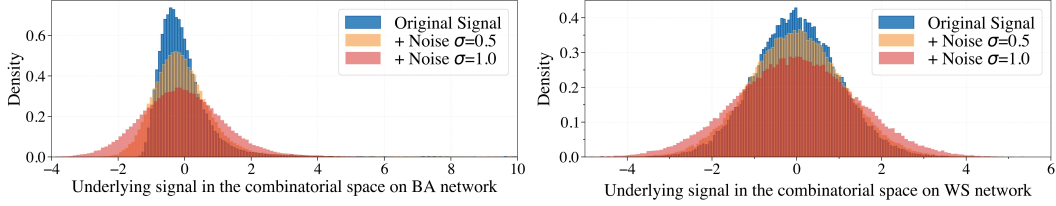



Figure 19: Density of underlying signals in the combinatorial space at different noise levels.

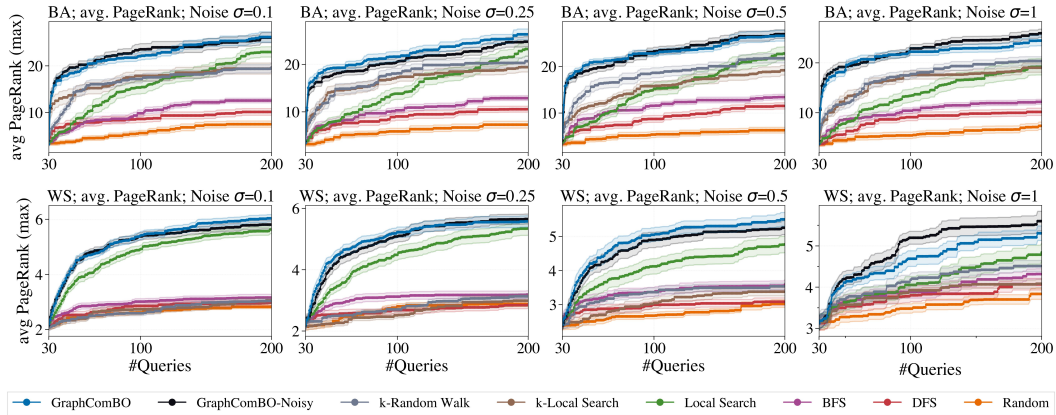



Figure 20: Maximizing avg. PageRank ($k = 8$) on BA and WS at different noise levels.

where the expectation and variance of the transformed function $\tilde{f}(S)$ are:

$$\mathbb{E}[\tilde{f}(\mathcal{S})] = 0 \tag{10}$$

$$Var(\tilde{f}(\mathcal{S})) = \frac{1}{k} Var\left(\sum_{i=1}^{k} PageRanks(\mathcal{S}_i)\right) \tag{11}$$

$$= \frac{1}{k} \times k \times Var(PageRanks(v)) \tag{12}$$

$$= 1 \tag{13}$$

Now, we can simply add random Gaussian noise to $\tilde{f}(\mathcal{S})$. Specifically, we consider $\epsilon \sim N(0, \sigma_\epsilon^2)$ with σ_ϵ at $[0.1, 0.25, 0.5, 1]$, where the level of noise can be directly estimated since both the underlying function and noise now have a mean of 0 and a standard deviation of 1. In addition, we further plot the estimated density of the original and noisy signals in Figure 19 to intuitively visualize the difference, which is done by randomly sampling 10^5 observed values in the combinatorial space $\binom{N}{k}$.

GraphComBO-Noisy. To better tackle the noisy observations, we implement *GraphComBO-Noisy*, which uses the best posterior mean across both visited and non-visited combo-nodes within the combo-subgraph as the new center, and then compare its performance to the original method which is guided by the observation. The results in Figure 20 show that the original method GraphComBO is robust to the noisy observations on both networks at different noise levels from $\sigma = 0.1$ to $\sigma = 1$. Compared with GraphComBO-Noisy, the observation-guided method performs comparably in most cases, except for a very noisy setting when $\sigma = 1$ on WS networks, where we can observe a clear advantage from the method guided by posterior mean, and it can be explained as follows.

Unlike classical discrete combinatorial functions of independent variables, the underlying functions in our problems are highly related to the graph structure. For example, BA networks are known for rich structural information due to the scale-free characteristics (i.e. node degree is exponentially distributed), which makes the distribution of the original signal heavily right-skewed with extreme values even after standardization (Figure 19 Left). By contrast, the WS small-world network (randomly rewired from a ring) has more homogeneous node degrees, and thus the original signal will be

27

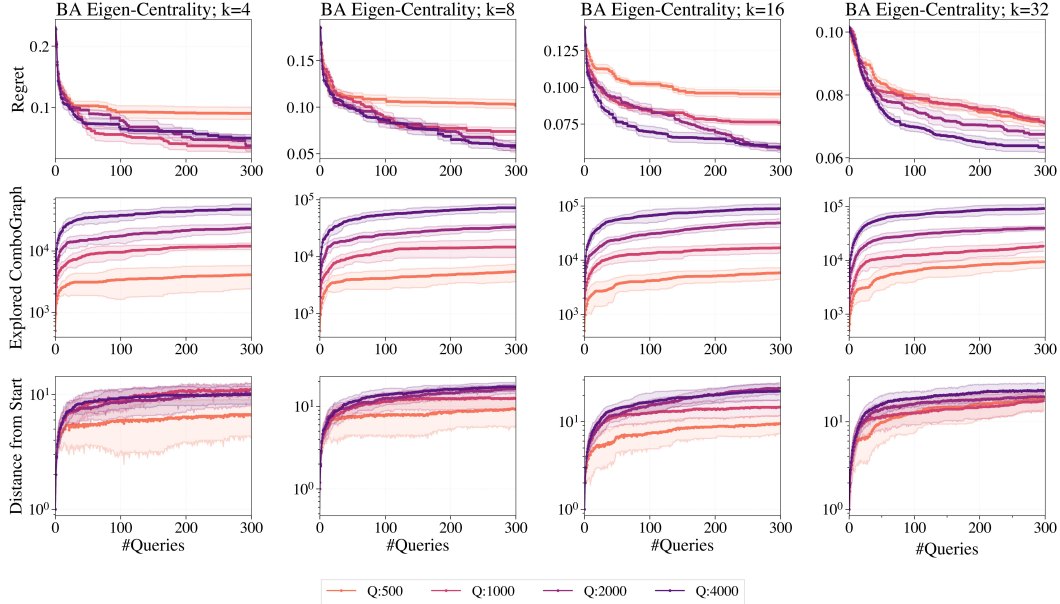



Figure 21: Ablation study of $Q = [500, 1k, 2k, 4k]$ with a fixed `failtol` = 30 on BA network.

more normally distributed after standardization (Figure 19 Right). Therefore, the noise level (even at $\sigma = 1$) is less significant on BA networks when the algorithm finds the promising region, whereas on WS networks at $\sigma = 1$, just as the reviewer described, we can see the algorithm is "misguided" by the observations compared to the posterior mean when the signal is highly corrupted.

K Ablation Studies on Hyperparameters

Lastly, we present the ablation study of BO's two hyperparameters: the combo-subgraph size Q and the tolerance of continuous failures `failtol`, and analyze their influence on BO's performance when using the best query location as `restart_method` over BA network and WS network.

Q   To analyze the impact of Q on BO's results, we fix `failtol` = 30 and vary Q at [500, 1000, 2000, 4000] and present the results in Figure 21 for BA and Figure 22 for WS. Overall, we can see that a larger Q will lead to better performance in most situations, this is because we are starting with a random location in the combinatorial space and thus exploration is more important than exploitation. The explored combo-graph size and the distance of combo-subgraph center from the start also validate this interpretation, where a larger Q generally leads to a larger exploration region that contains more distant nodes of higher querying values, therefore leading to better search performance. In addition, we can also observe that as k increases, the performance gain from larger Q becomes more salient, which further corroborates the statements in Section §4.

`failtol`   We analyze the influence of `failtol` on BO's performance by using a fixed $Q = 4000$ and vary `failtol` at [10, 30, 50, 100], where the results are presented in Figure 23 for BA and Figure 24 for WS. We can observe that despite a small `failtol` is able to explore a larger region in the combinatorial space, the performance gain from this behavior is limited. We make the following explanations. Since the underlying functions used on both graphs (average eigenvector centralities) are rather smooth, restarting with a random location usually leads to worse overall search performance, and we adopt the best query location as our `restart_method`. As a result, even if the algorithm restarts more frequently, it is still exploiting the regions around the same center. Nevertheless, we argue that with a different non-smooth underlying function, or when having a good initial location, a small `failtol` may have an advantage over the larger ones for its heavier exploitation behavior, and we leave this analysis to future work.

28

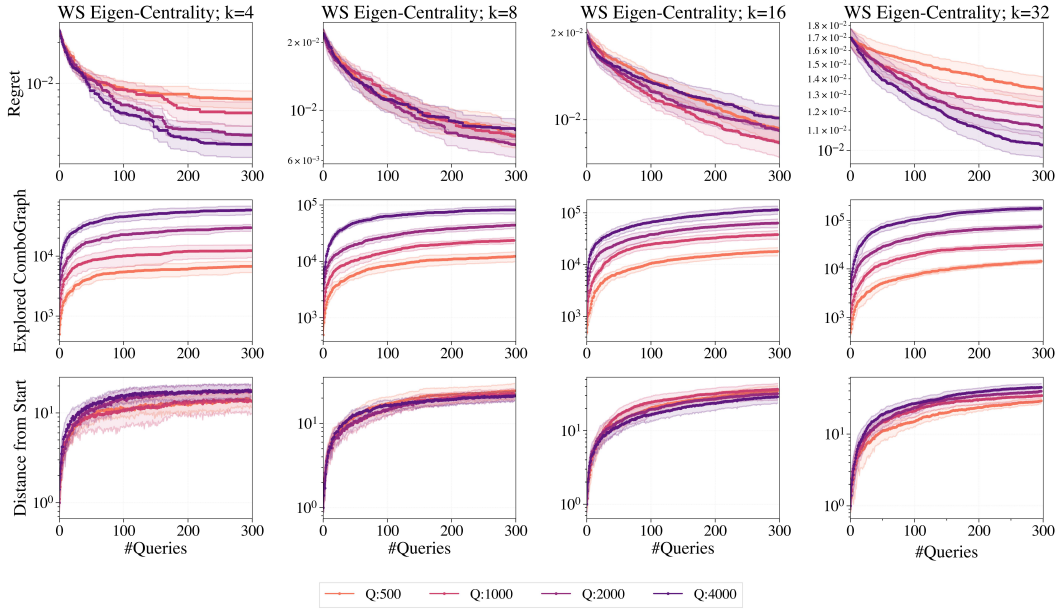



Figure 22: Ablation study of $Q = [500, 1k, 2k, 4k]$ with a fixed `failtol` $= 30$ on WS network.

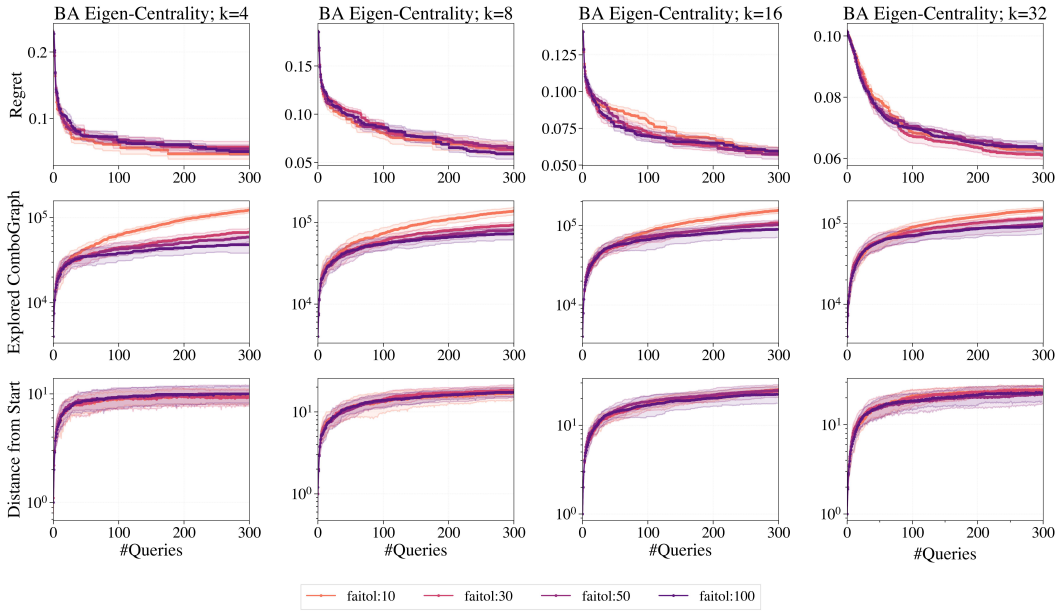



Figure 23: Ablation study of `failtol` $= [10, 30, 50, 100]$ with a fixed $Q = 4000$ on BA network.

29

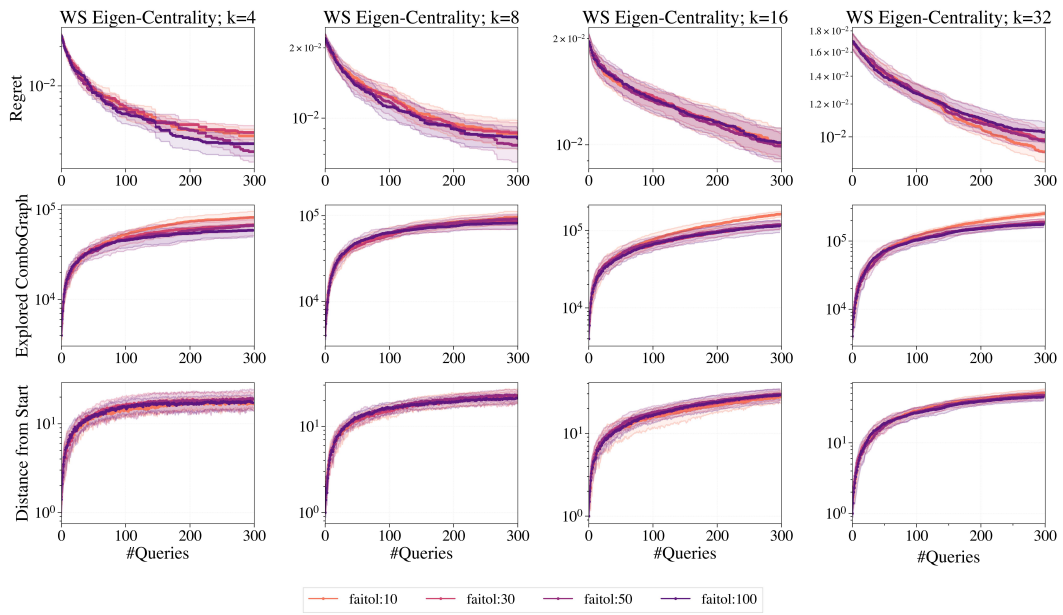



Figure 24: Ablation study of `failtol` = $[10, 30, 50, 100]$ with a fixed $Q = 4000$ on WS network.

30

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We clearly state our contributions in the Abstract and the last paragraph of the introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discussed our limitations on the subset size k throughout the paper, and we also provide a dedicated section §5 in the appendix for limitation and future work.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

31

Answer: [Yes]

Justification: We provide proofs for our Lemma in §D

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide detailed experimental settings as well as the hyper-parameters for reproducing our results in Appendix §B.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

32

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We open-sourced our code in an anonymous [LINK].

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We specify all the detailed settings in Appendix §B for results reproduction.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We report our results as the mean and stand error from 20 runs with different random seeds for all experiments in §4.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

33

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We specify the hardware details and running time information in §B.

Guidelines:
- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

Answer: [Yes]

Justification: We confirm that this research work conforms with the NeurIPS Code of Ethics.

Guidelines:
- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: We discuss the potential broader impact in §5.

Guidelines:
- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

34

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: We believe this research work requires no safeguard.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We confirm that we have properly cited and credited the original owners of the code in our implementation and do not violate their license and terms of use.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

35

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We have included instructions on how to run our codes with detailed comments inside the code explaining their functions.

    Guidelines:
    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: There is no crowdsourcing nor experiment with human subject involved our research.

    Guidelines:
    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: No crowdsourcing nor experiment with human subjects are involved our research.

    Guidelines:
    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

36