World Wide Web (2023) 26:1157–1179
https://doi.org/10.1007/s11280-022-01081-8

Self-supervised end-to-end graph local clustering

Zhe Yuan[1]

Received: 20 May 2022 / Revised: 19 June 2022 / Accepted: 24 June 2022 /
Published online: 8 July 2022

Abstract
Graph clustering is a central and fundamental problem in numerous graph mining applications, especially in spatial-temporal system. The purpose of the graph local clustering is finding a set of nodes (cluster) containing seed node with high internal density. A series of works have been proposed to solve this problem with carefully designing the measuring metric and improving the efficiency-effectiveness trade-off. However, they are unable to provide a satisfying clustering quality guarantee. In this paper, we investigate the graph local clustering task and propose a `End-to-End` framework `LearnedNibble` to address the aforementioned limitation. In particular, we propose several techniques, including the practical self-supervised supervision manner with differential `soft-mean-sweep` operator, effective optimization method with `regradient` technique, and scalable inference manner with Approximate Graph Propagation (AGP) paradigm and `search-selective` method. To the best of our knowledge, `LearnedNibble` is the first attempt to take responsibility for the cluster quality and take both effectiveness and efficiency into consideration in an `End-to-End` paradigm with `self-supervised` manner. Extensive experiments on real-world datasets demonstrate the clustering capacity, generalization ability, and approximation compatibility of our `LearnedNibble` framework.

Keywords Graph local clustering · Self-supervised learning · Generalized PageRank · Conductance

1 Introduction

Graph is a powerful framework to model the complex relations and interactions of our world [1–4]. The analyzing methods on graph have become fundamental and crucial. Graph clustering serves as an admiring and essential technique for wide-range applications, including community detection [5–7], image segmentation [8–11], protein grouping [12, 13] and especially spatial-temporal system [14–16], thus has drawn increasing attention during the recent years in computer science together with plenty of various applied

✉ Zhe Yuan
zhe.yuan@mor.org.cn

1 The School of Information, Renmin University of China, Zhongguancun St. 59, Haidian 100872, Beijing, China

research areas. However, the graph clustering faces efficiency and effectiveness challenges from both practical and theoretical aspects.

Locality One fundamental consensus towards efficiency challenge is that the graph clustering methods should be local w.r.t. some given seed node [17–19]. The locality has two aspects of meanings: process and result, which are usually coupled together for their implicit consistency. The process aspect means the graph clustering algorithm should only access the data in the neighborhood of the given seed node. The requirement for the result being local means the graph clustering algorithm should output the result nodes set in a small region around the seed node.

Framework Spielman and Teng [17, 20] are the first to study the graph local clustering problem. They propose a two-phase framework `Nibble` to guarantee the locality and performance of the graph clustering algorithm based on the analysis of Lovász and Simonovits [21, 22]. We introduce the important `Nibble` framework in detail in Section 2.3 and go through it quickly here. In the first phase, a power series $\left\{\mathbf{P}^k\vec{1}_s\right\}$, which represents the transition probability from seed node s to other nodes with k steps, is calculated. In the second phase, the standard `sweep` operation is conducted to output the node set with the first or global minimal conductance as the clustering result. The effectiveness of Nibble is guaranteed by the theoretical bounds on cluster quality [18, 23] and illustrated by empirical evaluations on various real networks [24–27].

Efficiency `Nibble` has been improved from two main aspects: measuring metric designing and measurement computing, leaving the `sweep` process as the standard and static module. Andersen, Chung, and Lang [18] propose the algorithm `PRNibble` by assembling the K-hop transition probabilities with the weight formed as $\alpha(1-\alpha)^k$ determined by the teleport constant α, which is a widely used node proximity metric named Personalized PageRank [28]. Besides, `PRNibble` [18] also provides an efficient local operation to compute it named `PR-Push` and achieve a better efficiency and effectiveness guarantee. Chung [24] extends the `PRNibble` with a physics-innovated metric called Heat Kernel PageRank(`HKPR`) whose weights are formed as $\frac{e^{-t}t^k}{k!}$, where the parameter t is the temperature constant controlling the distribution shape. With the improved theoretical bound, Chung and Simpson [29] propose a randomized algorithm `ApproxHK` with sampling random walks weighted by the heat kernel coefficients to get the approximated `HKPR` and propose a sub-linear solution `ClusterHKPR` for the graph local clustering problem. To optimize the `HKPR` computation, Kloster and Gleich [25] attempt to get rid of the heavy and randomized Monte-Carlo process in the `ApproxHK` by forming the computing problem as a linear equation solving problem. They propose an efficient deterministic algorithm `HK-Relax` to solve the proposed linear equation by using the coordinate relaxation technique and get a faster and better algorithm for graph local clustering. Recently, Yang et al. [30] point out that though the absolute error bound in `HK-Relax` is not the best choice for the graph local clustering task, under the perspective of the `sweep` process on the measure with degree normalization. Based on this observation, they propose the algorithm `TEA` to approximate the `HKPR` vector from the seed node with relative error bound, and achieve a better efficient-effectiveness trade-off. Wang et al. [27] optimize the `Push` operation in several works mentioned above by introducing the randomization and propose an efficient graph propagation framework `AGP`. `AGP` can simulate any weighted message passing schema and achieve state-of-the-art graph local clustering task performance with the `HKPR` weights.

Evaluation criterion Though the techniques mentioned above achieve better efficiency and effectiveness-efficiency trade-off, it is ambiguous and difficult for us to evaluate their effectiveness for the following two main reasons: 1) The algorithms are not developed to optimize the effectiveness, and 2) the effectiveness criterion and metric are not well defined. We first talk about the latter evaluation criterion problem by introducing several metric of cluster quality. Girvan and Newman [1] bring the concept of the cluster into graph research to represent the nodes set in such graph organized into internal densely linked but external loosely connected groups, which is also known as communities in network science [3]. To characterize the intuition of cluster concept, several scoring functions defined on graph structure have been proposed [6, 31–35], among which the `modularity` and the `conductance` are two essential metrics. `modularity` [34] metric evaluates the difference between the sub-graph with regard to the cluster nodes and the random graph with the same statistic properties. The `conductance` [36–38] metric directly describes the initial concept of the cluster with the Raleigh quotient form, which is formalized in Definition 3. Yang and Leskovec [39] compare a series of existing metrics on 230 real-world graphs with ground-truth cluster labels by defining sense-making and convincing criterion on goodness and robustness. They point out the `conductance` metric achieves the best performance during structural defines for graph clusters. Since high-order structures have advantages on revealing the real communities [40, 41], `motif-conductance` has been proposed recently and studied with a series of work [42–46]

Besides the cluster quality metrics based on graph topology, Emmons and Kobourov [47] propose the concept of information recovery metrics based on the Shannon entropy [48] defined on the ground-truth label of the graph cluster, including Adjusted Rand Index (`ARI`) [49] and Normalized Mutual Information (`NMI`) [50].

Optimization purpose In practice, graph local clustering works always take `conductance` and `F1-Score` (or `ARI`, `NMI`) as the criterion, seeing whether they could get both of them improved. Meanwhile, they generally achieve only one of them, which always be the information recovery one [25, 26, 51], making the result less convincing. Another kind of performance criterion is the trade-off between efficiency and effectiveness, which compares the cost to achieve the same effectiveness or the measuring score with the same algorithm cost, and adopted by the mainstream researches [27, 30]. However, even though the effectiveness would get better along with the algorithm running, we have no sense about the effectiveness, e.g., `conductance` or `F1-Score`, the algorithms ought to achieve and the appropriate time to stop the algorithm. The fundamental works of graph local clustering [18, 20, 29] suffer from a similar problem. The theoretical bounds with form $O(\sqrt{\Phi})$ may be less meaningful for the specific application situation whose purpose is finding the cluster with the best measuring score but not just with some bound guarantee.

Effectiveness To explore the solution of the evaluation dilemma and bring effectiveness into clustering algorithms, a series work [26, 51–53] focus on the measuring metric designing in the first phase of `Nibble` and push a step forward in both theoretical and applied areas. Kloumann, Ugander, and Kleinberg [52] regard the power series of transition probabilities as node features relevant to cluster and make the assembling weights a kind of linear classifier that digests these features to get the `GPR` space separated. They point out that `PPR` with a proper choice of the teleport constant α corresponds to the optimal classifier under Stochastic Block Model (`SBM`) [54] with the mean filed assumption [55]. Li, Chien, and Milenkovic [26] generalize the result by relaxing the mean-field assumption and analyzing the convergence of transition probabilities to their mean-field values and propose a new measure form with $\frac{\theta^k}{(\theta^k+\phi)^2}$ called Inversed PageRank (`IPR`) for their slower decay speed compared with `PPR`. Another inspiring work upon effectiveness is the Time-Dependent Personalized PageRank (`TDPR`) provided by Avron and Horesh [51]. Besides the new `GPR` structure, they also propose a new quality metric to evaluate the effectiveness of algorithms based on the differences between the results produced by different methods. They show that the proposed `TDPR` measure performs differently from the popular `PPR` and `HKPR` and could cooperate with the existing measures.

1.1 Motivations and challenges

Though a series of measuring metrics have been proposed to achieve better effectiveness, the problem of parameter selection under given metric is still challenging, such as the teleport parameter α in `PPR`, the temperature parameter h in `HKPR`, and the decaying parameters θ,ϕ in `IPR`. Though each work could choose the best parameters for its own purpose, they actually have no idea how to tune the parameter to get a better result, which leads the literature usually share the same parameters with some original work to take fair comparisons. Yang et al. [30] share their consideration of the parameter choice for their `TEA` algorithm and claim the importance of choosing the appropriate parameter for specific graph task. Klicpera, Weißenberger, and Günnemann [56] try to explore the best `GPR` weighting parameters for the graph local clustering task with the adaptive diffusion paradigm [57], which is designed for the link-prediction tasks. However, they find it performs worse than specific `PPR` and `HKPR` used in most other works. Li et al. [53] set up a `End-to-End` learning framework Gumbel-Softmax-based Optimization (`GSO`) to solve the optimization problems on graph, with the help of the `Gumbel-Softmax` technique, which could provide gradient to the sampling operations approximately. Though the framework is designed for all graph optimization problems, `GSO` would lose its ability to the massive graphs since the supervision signals in the graph learning problems are always sparse and `GSO` has $O(n)$ parameters to train.

Motivations To this end, we summarize the aforementioned statements and analysis with several questions as our motivations to conduct this research.

- Though the capacity of the `GPR` has been studied a lot, has it been fully demonstrated with the existing fixed parameters?
- To achieve better effectiveness, are there measures appropriate for different circumstance, and how can we design them?
- To achieve better efficiency, can we avoid conducting the grid-searching operations in graph clustering problem?
- To be scalable, can we take advantages of the existing techniques, e.g., approximation or randomization?

Challenges There are several fundamental challenges for designing and solve the problems described above. We give a brief description here and answer them in Section 3.

- How to deal with the discreteness of the `sweep` phase of the `Nibble` and make it differentiable to play a role in the `End-to-End` framework?

- How to make the conductance metric a proper supervising signal to provide the appropriate gradient to the training process?
- How to get the End-to-End model trained as desired?
- How to use the trained model to infer the clustering result?
- How to make the framework compatible with the present scalable graph local clustering algorithms?

Motivated by these inspiring questions, we focus on the measuring metric designing problem under the Nibble two-phase framework and develop a End-to-End learning framework LearnedNibble to efficiently and effectively optimize the graph local clustering target. More specifically, we model the measuring metric designing problem as the parameter selection task under the GPR form, whose capacity on the graph local clustering task has been proven in both theoretical and practical areas. We take graph topology $G = (V, E)$ and the seed node u as input since the relation between the semantic context on graph and the cluster structure is beyond our scope. We evaluate the algorithm performance with the conductance metric because conductance is consistent with the initial and natural definition of the cluster and performs well in both experimental and applied circumstances. By solving these non-trivial challenges in an integral framework, we bring a new perspective and framework to the graph local clustering task.

1.2 Our contributions

We present an in-depth study on Nibble-based graph local clustering task with conductance as the cluster quality metric and make the following contributions.

Supervision manner We design a differentiable learning-based soft-mean-sweep operator in a self-supervised manner to guide the training process.

Optimization mechanism We explore the appropriate optimization mechanism for the graph local clustering task and propose the regradient technique to conduct the optimization.

End-to-end framework We model the effectiveness problem of graph local clustering as a learning task w.r.t. GPR weighting parameters, and propose a End-to-End framework named LearnedNibble based on the soft-mean-sweep and the regradient technique, which can adaptively raise the cluster with best conductance score on different graphs.

Capacity and compatibility We illustrate the capacity of the GPR family and our LearnedNibble framework by conducting extensive experiments on the standard benchmarks of graph clustering tasks. We show that LearnedNibble gets the better effectiveness against all existing and commonly-used measuring metrics, e.g., PPR, HKPR and IPR, in all datasets. Moreover, the advantage of LearnedNibble is still kept with all levels of approximation, allowing it to combine with any approximated local clustering framework.

Scalability and practicality We show that the clustering manner obtained from our LearnedNibble can generalize to the other nodes, whether they are in the same cluster as the seed node or not, with just a slight performance reduction. The generalization ability of LearnedNibble makes it scalable to massive data circumstances and practical in diverse graph-based tasks, including Graph Visualization and Graph Neural Networks.

1.3 Paper organization

The rest of the paper is organized as follows. We introduce some basic notations and important techniques in Section 2, We present LearnedNibble framework in Sections 3. We evaluate the clustering capacity, generalization ability and approximation compatibility of our framework in Section 4. Finally, Section 5 discusses several interesting observations and shares some ideas and Section 6 concludes the paper.

2 Preliminaries

Before deriving the LearnedNibble framework in detail, we first introduce several important notations and techniques, and finally formalize the problem we investigate in this work.

2.1 Basic terminology

Let $G = (V,E)$ be an undirected and unweighted graph, where $V = \{v_1,v_2,...,v_n\}$ denotes the node set with size n, and $E = \{e(u,v)|u,v \in V\}$ denotes the edge set with size m. We use $d(u)$ to denote the node v's degree, and use vector $d = \{d(u),u \in V\}$ to represent degree corresponding to each node. We use $\mathbf{A}$ to denote the adjacency matrix of G, and $\mathbf{A}(i,j) = \mathbf{A}(j,i) = 1$ if and only if we have $e(v_i,v_j) \in E$. Let $\mathbf{D}$ be the degree matrix of G with $\mathbf{D}(i,i) = d(v_i)$. Besides, the transition probability matrix (a.k.a random walk transition matrix or random walk transition probabilities) for G is represented by $\mathbf{P} = \mathbf{D}^{-1}\mathbf{A}$. Accordingly, $\mathbf{P}^k$ denotes the k-th order transition probability matrix, $\mathbf{P}^k\vec{1}_s$ denotes the transition probabilities of the k-hop random walk started from seed node s. The notations used frequently in this work are listed in Table 1.

2.2 Generalized pagerank

This part introduces the measuring metric used in this work.

Definition 1 (L-hop Transition Probability Sequence) Given a graph G and the seed node s, the transition probability from s to other nodes $u \in V$ with k-steps can be computed as: $p^k(s,u) = \mathbf{P}^k(s,u)$. By putting nodes together we get the k-hop transition probability vector of s, i.e., $p^k(s) = \mathbf{P}^k\vec{1}_s = \left\{p^k(s,u)|u \in V\right\}$. The L-hop transition probability sequence is defined as the sequence of k-hop transition probability vector with the random walk length k ranging from 1 to L with form:

$$\pi^L(s) = \left\{p^k(s)|k \in [1,L]\right\}. \tag{1}$$

Definition 2 (Generalized PageRank) Given the L-hop transition probability sequence $\pi^L(s)$ of seed node s on graph G, the Generalized PageRank with the weighting vector w is defined as:

Table 1 Basic notations

Notation	Description
$G = (V,E)$	undirected connected graph with the set of vertices V and edges E
d	the degree vector of vertices V of the graph G with length n, whose u-th element is the degree of node u
$\mathbf{A}$,$\mathbf{D}$,$\mathbf{P}$	the adjacency matrix, diagonal degree matrix, transition matrix of G
L	The maximum number of hops during performing push operations from the seed node, set by 50 in this paper
π^L	the L-hop transition probability sequence of G, see Definition 1
w	the weight vector which assemble the L-hop transition probability sequence
$\mathbf{gpr}_w, \mathbf{gpr^d}_w$	the assembled L-hop transition probability vector, and its degree-normalized version
$\Phi(C)$	the conductance of the cluster C

$$\mathbf{gpr}_w^L(s) = w \times \pi^L(s) = \sum_{k=1}^{L} w_k \cdot \mathbf{P}^k \vec{1}_s. \tag{2}$$

We may omit the seed node flag s and the range flag L in the expressions and use π and $\mathbf{gpr}_w$ in brief.

2.3 Graph local clustering

A cluster in G is a node set $C \subset V$ and its quality is measured by a given criterion. We use the commonly-used `conductance` criterion in this work.

Definition 3 (Conductance) Let $G = (V,E)$ be a undirected, unweighted graph. The volume of a node set $C \subset V$ is defined as $\mathrm{vol}(C) = \sum_{u \in C} d(u)$. The edge boundary of a node set C is defined as $\partial(C) = \{e(u,v) | u \in C, v \notin C\}$. The conductance of a node set C is defined as:

$$\Phi(C) = \frac{|\partial(C)|}{\min(\mathrm{vol}(C), \mathrm{vol}(G \setminus C))}. \tag{3}$$

We introduce the `Nibble` two-phase framework, which is the fundamental framework of graph clustering tasks, by formally introducing each phase of it.

Definition 4 (`Measure`) Given the graph G, seed node s and the measuring metric $\mathscr{M}$, we use the $\mathscr{M}$ to measure the proximity score of all nodes towards s on graph G and output the measuring score vector $q = \mathscr{M}(G, s)$.

Definition 5 (`Sweep`) Given the measuring score vector q and the quality scoring function $\mathcal{S}$. Let $c = (v_1, ..., v_n)$ be an ordered sequence of the nodes such that $\frac{q(v_i)}{d(v_i)} \geq \frac{q(v_{i+1})}{d(v_{i+1})}$. We scan the sequence and make the top-j elements a candidate set C_j when visit j-th element. We use $\mathcal{S}$ to evaluate the quality of the candidate set sequentially, and outputs the C_* with best score, i.e., smallest conductance in this work, $\mathcal{S}(C_*) = \mathcal{S}_*$ as the result.

The sweep phase is demonstrated by Algorithm 1.

```
Algorithm 1: Sweep
    Input: The graph $G = (V, E)$, The measuring score vector $q$.
    Output: The proposed cluster $C_*$ with its score $S_*$.
1   $c \leftarrow sort(q)$ ;
2   $C = \emptyset$ ; $S_* = 1$ ;
3   $S \leftarrow \Phi$ for each node $v \in c$ do
4       $C = C \cup \{v\}$ ; $S = \mathcal{S}(G, C)$ ;
5       if $S \leq S_*$ then
6           $C_* \leftarrow C$ ; $S_* \leftarrow S$;
7   return $C_*, S_*$.
```

2.4 Approximate graph diffusion

The approximate graph diffusion(AGP) [27] framework shows a great capacity to handle the massive data circumstance. We make it a basic module in LearnedNibble for scalability sake. AGP takes an undirected graph G, a seed node s, a propagation range level L, a weighted sequence w and a error guarantee parameter ϵ as input, outputs the estimated propagation vector which achieves both theoretical approximate guarantee and near-optimal running time complexity. In our settings, we make the weight vector w an all-ones vector to get the estimated L-hop transition probability sequence $\hat{\pi}^L$ from the AGP process as the input of our LearnedNibble.

2.5 Problem formulation

With taking the effectiveness, efficiency and scalability into consideration, we formalize the problem investigated in this work as the End-to-End d approximate conductance optimization task described as follow.

Definition 6 (End-to-End Approximate Conductance Optimization) Given the graph G, seed node s, propagation range level L, error guarantee parameter ϵ. The estimated L-hop transition probability sequence $\hat{\pi}^L$ with absolute error ϵ is raised from the AGP. We follow the Nibble two-phase framework with keeping the sweep phase fixed as a standard cluster proposition process based on measuring score vector q, focus on finding the appropriate measuring metric $\mathscr{M}$ in Generalized PageRank form, i.e., $w \times \hat{\pi}^L$, to optimized the conductance of the proposed cluster, in an End-to-End d manner.

3 The framework

This section introduces our LearnedNibble framework with dealing with the challenges mentioned in Section 1.1 and to solve the problem defined as Definition 6.

3.1 Input data

The input data is not only the material on which our training process is based but also is the query task for which our model should take responsibility. We use the approximated result output by the AGP under some error guarantee parameter ϵ as our input data to make our framework compatible with approximation and scalable on massive graphs.

3.2 Trainable parameters

We model the $\mathscr{M}^{L}_{GPR}$ used in the Measure phase of LearnedNibble as an assembling method of estimated L-hop transition probability sequence $\hat{\pi}^{L}$ with trainable weighting parameters as:

$$\mathcal{M}^{L}_{GPR}(\pi) = w \times \hat{\pi}^{L} = \mathbf{gpr}_{w}. \tag{4}$$

Therefore, the parameter amount of LearnedNibble is L rather than $O(n)$ [53].

3.3 Supervision manner

As mentioned in Section 1.1, the sweep phase described in Section 5 is in grid-search manner and thus is discrete and not differential inherently, which brings challenges to achieve the desiring End-to-End d framework. With a careful investigation of the sweep phase, we divide the integral sweep apart into three operations, which are conducted in turn but coupled with each other in an ingenious way, namely the loop, selection and evaluation. We analysis these operations carefully in the following part to better introduce our intuition and solution.

3.3.1 Loop

The loop operation sequentially visits each element along the measuring score vector and conducts the following selection operation to guarantee the best result within all n cluster candidates. The loop operation makes the algorithm avoid the combinatorial complexity by reducing the check operation times from the Bell Number with parameter n to the n and provides the admiring locality. Nevertheless, the brute-force mechanism within the loop operation binds itself to the disappointing discreteness and makes it incompatible with the End-to-End manner. Therefore, questions come in two aspects. 1) How to activate the selection operation to get the candidate node sets? 2) How to guarantee the performance? We give our answers to both questions in the following part.

3.3.2 Selection

Towards the questions above, we present two of our several trials here, one of which finally forms the LearnedNibble.

Sharp-drop modeling We notice that Andersen and Chung [23] propose an powerful statement about the sweep phase, saying that whenever there is a sharp drop in the rank defined by a personalized PageRank vector, the location of the drop reveals a cut with small conductance. Inspired by this observation, we try to model the selection operation with a trainable parameter Δ. We expect it separates the measuring score vector into two parts, corresponding to the cluster and the rest. Unfortunately, this proposal suffers from the absence of the loop operation and the flexibility of the GPR measure $\mathscr{M}^L_{GPR}$ in Section 3.2. As a consequence of the first one, we cannot keep the learned Δ with a reasonable value which surely should be in the measuring score range, despite diverse training techniques or regularizations. Besides, we lose the connection between the GPR measuring score and the parameter Δ even within two consecutive learning epochs, making the two-stage training mechanism fail. Because it makes no sense to expect the best separation method for one score vector suits another well, as they may vary widely.

Self-supervising To handle it, we first revisit the Nibble framework to find out the most essential information covered by it. Though the performance seems to be related to the measure result and some values like the sharp drop, we point out that the clustering capacity is mainly determined and represented by the order of measuring score sequence under the Nibble manner. Once the score of each node is fixed, the clustering result is almost determined. Besides, unlike most other methods, we are not pursuing to output the final result in just one sweep run, but exploring the appropriate measurement method for the specific task in the training process. Therefore, we don't have to ask for the exact evaluation on the measuring score sequence as the standard sweep does. We only need to provide some information to guide the measure $\mathscr{M}^L_{GPR}$ to achieve better cluster discovering capacity by adaptive adjusting its weight parameters. Thus, we propose the mean-sweep technique to provide a lower-bound of the clustering capacity of the measure $\mathscr{M}^L_{GPR}$ by separating the score sequence into two parts based on mean of itself. We formally introduce the mean-sweep operation with the following definition.

Definition 7 (mean-sweep) Given the measuring score vector **gpr** with the measure $\mathscr{M}^L_{GPR}$. Let $\mathbf{gpr^d} = \frac{1}{d}\mathbf{gpr}$ be degree-normalized version of the **gpr**. We choose the nodes whose normalized measuring score is above the mean value, i.e.,

$$C_* = \left\{v_i | \mathbf{gpr^d}(v_i) \geq \bar{\mathbf{g}}\right\}, \bar{\mathbf{g}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{gpr^d}(v_i), \tag{5}$$

to be the cluster result.

Even though the mean-sweep only provides one clustering result among many possible selections, it is sufficient to guide the training process. We illustrate this statement with the experiment results in Section 4.

Although we have already stepped forward by providing a solution to the selection dilemma within the learning mechanism, the discreteness challenge still exists as the result proposed by mean-sweep is also a node set, which is discrete and blocks the gradient propagation. It leads us to the evaluation problem which is rather trivial in the standard sweep operation.

3.3.3 Evaluation

Following the same principle in the mean-sweep technique, we use the Sigmoid operator, which is widely used in the Machine Learning areas, to make the score above mean close to 1 and make the other close to 0. The activating operation here is not for bringing the system non-linearity but is an approximation of the discrete set selection result. It plays a similar role as the `Gumbel-Softmax` operation in the `GSO` [53]. With this approximation, we propose the `soft-mean-sweep` module, the core element of our `LearnedNibble` framework.

Definition 8 (`soft-mean-sweep`) Given the measuring score vector **gpr** with the measure $\mathscr{M}^{L}_{GPR}$. Let $\mathbf{gpr^d} = \frac{1}{d}\mathbf{gpr}$ be degree-normalized version of the **gpr**. We normalize the $\mathbf{gp}r^d$ with its mean and use the `Sigmoid` operator to activate it, i.e.,

$$c = \sigma\left(\mathbf{gpr^d} - \frac{1}{n}\sum_{i=1}^{n}\mathbf{gpr^d}(v_i)\right) \tag{6}$$

and make it the approximate clustering result.

Loss We get the final supervision manner for `LearnedNibble` by putting everything together. We compute the `conductance` in the Raleigh quotient on the result output by `soft-mean-sweep` with the matrix operation, view it as the approximate reflection on the clustering capacity provided by the $\mathscr{M}^{L}_{GPR}$, and set it as the supervising signal (a.k.a the `loss`) of the learning framework, i.e.,

$$\begin{aligned}\psi &= \frac{c^{\mathrm{T}}(\mathbf{D}-\mathbf{W})c}{\min\left(c_A^{\mathrm{T}}\mathbf{D}c_A, c_{\overline{A}}^{\mathrm{T}}\mathbf{D}c_{\overline{A}}\right)},\\ c_A &= 0.5\cdot(1+c), c_{\overline{A}} = 1 - c_A.\end{aligned} \tag{7}$$

3.4 Optimization mechanism

With the supervision manner and loss function in hands, the most important thing is using the supervising signal to guide the training process. Several optimizers have shown their capacities to be the appropriate engine of diverse learning tasks, among which the `Adam` [58] is the most widely-used one. Though being successful in plentiful circumstances, the `Adam` does not work well as expected in our graph local clustering task. It always misses the better solutions and sometimes keeps the wrong direction for a long time. We suppose one reasonable explanation of this wired situation is that the graph clustering task naturally has many local optimums who are close to the best one, which makes the `Adam` strapped and misled.

Regradient To mitigate the problem, we propose the `regradient` technique to make the `Adam` optimizer focus more on the current step and avoid being affected by the former gradients.

Definition 9 (`regradient`) With a parameter r which controls the restart frequency, we reset the `Adam` optimizer every r epoch by clearing its accumulated gradients. Without losing the generality, we fix the r to be 10 in this work.

We will present the effectiveness of the `regradient` technique with an ablation experiment in Appendix 1.

3.5 Inference manner

The last but not least thing is obtaining the model from the training process and using it to do the inference. In our `LearnedNibble` framework, the model is the measuring method $\mathscr{M}^{L}_{GPR}$ with learned `GPR` weight parameters, and the inference result is the clustering node set.

Most machine learning tasks obtain the final trained model with the convergence and the `early-stop` technique. As for our graph local clustering task, it is unnecessary and unfair to ask the model to get converged. The reasons are twofold. 1) As mentioned in the supervision manner in Section 3.3, we use the lower-bound of the clustering capacity to guide the training process, and there may be a gap between the performance reported by the `loss` and the actual ability of the model. 2) Though we aim to avoid searching the massive possible cases, we still share the same solution space as the former combinatorial optimization problem. Thus, we propose the `search-select` manner to obtain the model from the `LearnedNibble`.

Definition 10 (`search-select`) Given the L-hop transition probability sequence π and training process of the `LearnedNibble` with T epochs, i.e., $\mathcal{R}^T = \left\{\mathscr{M}_1, ..., \mathscr{M}_T\right\}$, where $\mathscr{M}_i$ is the measuring method with trained weight vector w_i, i.e., $\mathscr{M}_i(\pi) = w_i \times \pi$. We compute the exact clustering capacity Φ_i of $\mathscr{M}_i$ by conducting the standard `sweep` operation on the measure result of each $\mathscr{M}_i$ as described in Algorithm 1. We select the $\mathscr{M}_*$ with the best Φ_* as the final model. We use the $\mathscr{M}_*$ obtained from the training process $\mathcal{R}^T$ on graph G to answer the query of any seed node $s \in G$.

3.6 Framework overview

We present our `LearnedNibble` framework in this part with Algorithm 2 and Algorithm 3. The `Initialization` module which has not been mentioned is described in Appendix 1.

Algorithm 2: LearnedNibble-Train

```
Input: The graph $G$, propagation range level $L$, error parameter $\epsilon$,
       seeds size $k$, initialization parameters $\Theta$, training budget $T$,
       regradient step $e$, learning rate $lr$.
Output: The clustering method $\mathcal{M}_G$ for graph $G$.
1  $\mathbb{M}_G \leftarrow \emptyset$ ;$seeds \leftarrow$ Sample$(V, k)$ ; Adam $\leftarrow$ Adam$[e]$ ;
2  for each node $s \in seeds$ do
3      $\mathbb{M} \leftarrow \emptyset$ ;$\mathcal{M}_0 \leftarrow$ Initialize$(\Theta)$ ;$\mathbb{M} = \mathbb{M} \cup \mathcal{M}_0$ ;
4      $\hat{\pi} \leftarrow$ AGP$(G, s, L, \epsilon)$ ;
5      for epoch $s \in [0, T]$ do
6          gpr $\leftarrow \mathcal{M}_{epoch}(G, \hat{\pi})$ ;
7          $\psi \leftarrow$ soft-mean-sweep(gpr) ;
8          $\mathcal{M}_{epoch+1} \leftarrow$ Adam$(\psi, \mathcal{M}_{epoch}, lr)$ ;
9          $\mathbb{M} = \mathbb{M} \cup \mathcal{M}_{epoch+1}$ ;
10     $\mathcal{M}_s \leftarrow$ search-select$(\mathbb{M}, \hat{\pi})$ ;$\mathbb{M}_G = \mathbb{M}_G \cup \mathcal{M}_s$ ;
11 $\mathcal{M}_G \leftarrow$ Mean$(\mathbb{M}_G)$;
12 return $\mathcal{M}_G$.
```

```
Algorithm 3: LearnedNibble-Infer
    Input: The graph $G$, propagation range level $L$, error parameter $\epsilon$,
           query node $s$, clustering method $\mathcal{M}_G$.
    Output: The cluster $C_*$ rooted at node $s$ and its conductance $\Phi_*$.
1   $\hat{\pi} \leftarrow \texttt{AGP}(G, s, L, \epsilon)$ ;
2   gpr $\leftarrow \mathcal{M}_G(\hat{\pi})$ ;
3   $C_*, \Phi_* \leftarrow \texttt{sweep}(G, \mathbf{gpr})$ ;
4   return $C_*, \Phi_*$.
```

4 Experiments

This section evaluates the performance of our LearnedNibble in three aspects: 1) the clustering capacity concerning conductance optimization, 2) the generalization ability from training seed nodes to the whole graph, 3) and the compatibility with the approximation. We report the key results here and discuss additional results in Appendix 1.

Datasets We conduct our experiments on commonly-used benchmark graphs with ground-truth labels, including DBLP, Amazon, PubMed, CiteSeer and Cora. The statistics of the datasets are listed in Appendix 1 with Table 4.

Metric We use the conductance as the evaluation metric and the optimization target of our framework since the information recovery metrics could conflict with our optimization purpose. We investigate the conductance of the ground-truth clusters in Figure 2 in Appendix 1.

Competitors We set the existing GPR instances with different specific weighting paradigm, like PPR, HKPR and IPR, as part of our competitors. Another competitor is the MEAN weighting operation since the result proposed by any method should be better than this trivial one. The last competitor we set for our LearnedNibble is the most recent GSO [53] as for its applicability for all graph optimization tasks. The considerations and the comparison methods are presented in Appendix 1.

4.1 Training settings

Training data We select 5 seed nodes from different clusters whose size is larger than 100 randomly from each graph to form the training seed node sets. We set the propagation range $L = 50$. We vary the approximation parameter ϵ in $[0, 10^{-4}, 10^{-5}, 10^{-6}]$. $\epsilon = 0$ means we use the exact L-hop transition probability sequence rooted at the seed node.

Initialization method We use the RAW weighting vector, which is a one-hot vector with the seed node index non-zero as the initial weight for the training process since the other initialization methods are our competitors. The analysis of the initialization sensitivity is presented in Appendix 1

Training method We set the training budget $T = 2{,}000$, the `regradient` step $e = 10$ and the learning rate $lr = 1$.

4.2 Clustering capacity

This section investigates the clustering capacity of `LearnedNibble` with no approximation. The results are presented with Table 2. It is surprising to see that the trivial `MEAN` beats all `GPR` instances in all datasets. Moreover, our `LearnedNibble` shows much better performance compared with all competitors. The `GSO` seems to learn nothing from the training process, and we will omit it for the following comparisons. The specific settings and detailed results are presented in Appendix 1.

4.3 Generalization ability

We present the generalization ability of `LearnedNibble` with no approximation in this part by reporting basic statistics of the conductances in test samples. The results are listed in Table 3. First, the final model obtained from `LearnedNibble` is useful as it gets even better performances when transferred to other nodes, no matter within the cluster or the whole graph. See the last column of Table 3. Then, the `mean` and the `std.` columns prove that the model achieves a satisfying performance wvector in our situatioith having strong confidence to find a cluster with relatively small `conductance`. We talk about some other interesting observations in Section 5.

4.4 Approximation compatibility

The approximation compatibility of `LearnedNibble` is presented in this section with four different approximation levels. Figure 1 combines all information together. We report 4 results of total 10 results(5 datasets and 2 generalizations for each) for the convenience sake, where the * represents the `in-cluster` generalization results. The other results are presented in Appendix 1. The x-axis is the approximation level with parameter ϵ. The y-axis is the `conductance` value. The thick horizontal line is the training results. The boxplot represents the transferring results. Though the clustering capacity and the generalization ability of the `LearnedNibble` would be weakened with the approximation level up, it is still rather satisfying for the most time since the `conductance` values are rather small with variance well-bounded.

Table 2 Clustering capacity

Dataset	PPR	HKPR	IPR	MEAN	GSO	GPR(ours)
DBLP	0.1104	0.1125	0.1307	0.0958[‡]	0.4985	**0.0380**[*]
Amazon	0.0566	0.0561	0.0828	0.0494[‡]	0.4977	**0.0159**[*]
PubMed	0.0830	0.0861	0.0990	0.0681[‡]	0.4995	**0.0391**[*]
CiteSeer	0.0363	0.0358	0.0482	0.0280[‡]	0.5016	**0.0187**[*]
Cora	0.0812	0.0792	0.0878	0.0685[‡]	0.4948	**0.0305**[*]

[‡] The values with ‡ are the best among baselines

[*] The values with * and in bold are the best results

Table 3 Generalization ability

Dataset	Train	Test-Mean	Test-Std.	Test-Max.	Test-Min.
DBLP	0.0380	0.0806	0.0396	0.1675	0.0102
DBLP*	0.0380	0.1705	0.0201	0.2016	0.1174
Amazon	0.0159	0.0184	0.0080	0.0406	0.0090
Amazon*	0.0159	0.0178	0.0090	0.0481	0.0062
PubMed	0.0391	0.1502	0.0965	0.3906	0.0367
PubMed*	0.0391	0.0727	0.0369	0.2266	0.0363
CiteSeer	0.0187	0.0417	0.0732	0.4444	0.0111
CiteSeer*	0.0187	0.0675	0.1045	0.3333	0.0050
Cora	0.0304	0.1019	0.0416	0.1949	0.0256
Cora*	0.0304	0.0804	0.0199	0.1188	0.0261

The rows with * are in-graph tests, and the others are in-cluster tests

5 Discussions

Though the experiments result in Section 4.3 and Appendix 1 shows positive evidence for sharing the similar clustering method for all nodes on the same graph, some weird but interesting phenomena have got our attention. 1) The in-cluster generalization performance is worse than the in-graph one in `PubMed` and `Cora`. 2) The performance gap between the in-cluster and in-graph is large in `DBLP` and `Amazon`. 3) We can get better results for

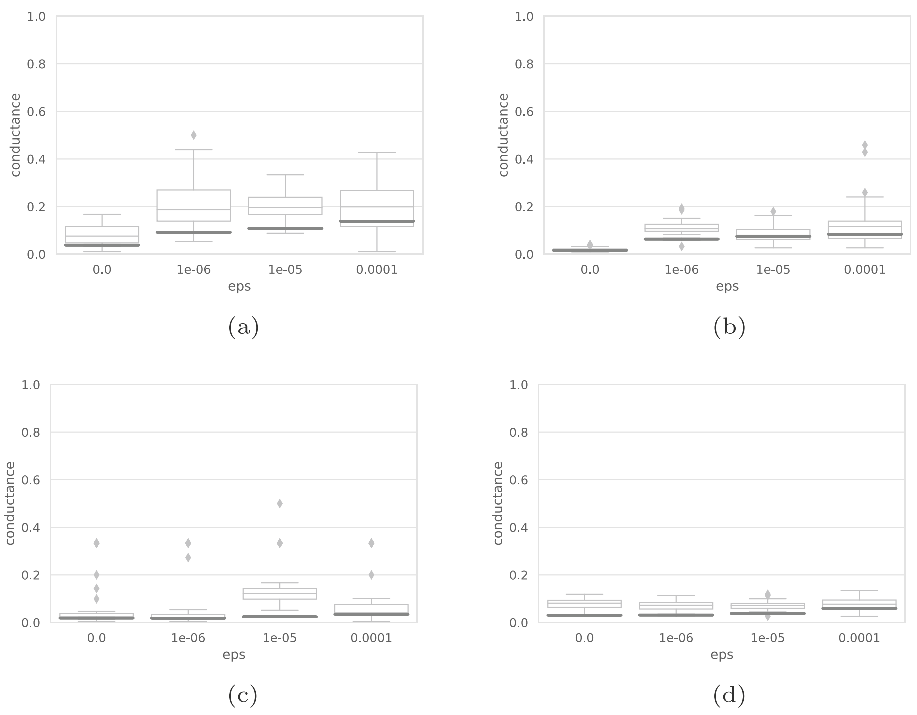

Figure 1 Capacity and ability with approximation. (**a**) DBLP* (**b**) Amazon* (**c**) CiteSeer (**d**) Cora

some nodes even not been optimized in nearly all situations. These observations get us to consider the basis of the generalization and the information attached to the graph.

Topology consistency The most critical assumption we should have to transfer one model to other situations is the consistency. As for graph clustering tasks concerning the topology structure optimization metric like `conductance`, the topology consistency of different parts of graph should be evaluated and checked firstly, which is another interesting question.

Data representativeness As described in Section 4.1, we only use 5 randomly chosen nodes as our training data, which may not be so representative for the graph. Thus, selecting suitable nodes as training data may be a fundamental problem.

Information-topology compatibility Recall that we have pointed out the conflict between topology-based metrics and information-based metrics in Section 1 and Appendix 1, the information attached to the graph as labels or other context on nodes or edges may provide somewhat different and independent information compared to the topology. As a result, the generalization in the so-called same cluster could be meaningless and even more challenging. Undoubtedly, taking advantage of both topology and information of graph is one of the most crucial but challenging problems in the graph mining area.

6 Conclusions

In this paper, we take in-depth research on the graph local clustering task and propose a novel learning-based framework `LearnedNibble` by solving a series of non-trivial challenges. To the best of our knowledge, `LearnedNibble` is the first one to take responsibility for the cluster quality and take both the effectiveness and efficiency into consideration in an `End-to-End` paradigm with `self-supervised` manner. Our experiments demonstrate that the clustering capacity of L-hop transition probability sequence is underestimated with only using the fixed weighting structures and parameters to assemble, and can be taken better advantage by our `LearnedNibble` framework. Besides the performance improvements on the cluster quality, our framework shows great generalization ability and approximation compatibility, making itself practical in many situations.

Appendix A: additional experiments

Data sources We obtain the `DBLP`, `Amazon` from the Stanford Network Analysis Project(`SNAP`) [59], and the rest from their original works [60, 61]. We present the basic information of the datasets used in our experiments in Table 4, and take a view of the conductances of the ground-truth clusters with Figure 2. We can see that the `conductance` of the labeling clusters are rather large, which should make the information-based metrics conflict with the structure-based metrics, as we note in the following part.

Competitor considerations Since the effectiveness challenge has not been studied much and little work targets the `conductance` metric as we do, the competitor of our `LearnedNibble` may not be any specific research result or algorithm. Besides, the work we

Table 4 Statistics of graph datasets

Dataset	*n*	*m*	*cmts*[*]
DBLP	317,080	1,049,866	13,477
Amazon	334,863	925,872.	75,149
PubMed	19,717	44,338	3
CiteSeer	3,312	4,732	6
Cora	2,708	5,429	7

[*] *cmts* is the community numbers

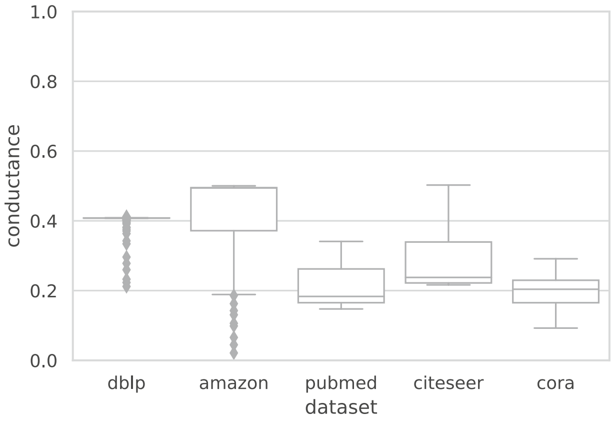

Figure 2 Conductance of the ground-truth of clusters

present here does not aim to beat any baseline but reveals the capacity of `GPR` measure family and explore the possibility and method to realize them, with being compatible to the mainstream approximate algorithms.

Comparisons For `GPR` instances, we evaluate them by grid-searching a bunch of parameters with 2,000 trials for each, which is also the training budget for `LearnedNibble`, and take the best performance as their clustering capacities. Specifically, we set the $\alpha \in [0, 1, 0.0005]$ for `PPR`, $h \in [1, 20, 0.01]$ for `HKPR`, $\theta \in [0, 1, 0.005]$ and vary the power of θ which determines the ϕ in $[1, 5, 20, 50, 100]$ for `IPR`. For `MEAN`, we directly compute its exact `conductance` by the standard `sweep` operation. For `GSO`, we set the training budget of 200,000 for it since it has much much more parameters to train.

A.1 Training details

We make the `LearnedNibble` have the full accessibility of the graph adjacency matrix in the training phase but keep the algorithm local in the inference phase as other computing-based graph local clustering algorithms. The reason we make the algorithm not thoroughly local is twofold. 1) First, we should use the whole graph data since the topology is integrated and should not be sampled as the data points in the Euclidean space. 2) Second, we are looking forward to seeing that the framework have a good generalization ability to the whole graph, which is the crucial character we may depend on to develop the scalability and practicality of `LearnedNibble` while making the algorithm local seems weird and maybe conflict with the purpose.

For the trainable weighting parameters, we normalize the weight vector w to be one-norm $\|x\|_1 = 1$ in the inference phase but keep it free in the training phase for numerical stability sake.

A.2 Clustering capacity details

Comparisons We report the average `conductance` of the 5 training seed nodes with the final model in each datasets with Table 2. The first 4 columns are the `GPR` family instances and the trivial `MEAN` pooling operation. The GSO column represents the `GSO` [53] framework. The last column with title GPR is our `LearnedNibble` framework.

Results with approximation in detail We report the results of different datasets in turn and list them with Table 5.

A.3 Generalization ability details

Comparisons To see more clearly, we report the generalization abilities of our `LearnedNibble` framework with competitors in two aspects. 1)In-Cluster: We do inference on

Table 5 Comparisons with approximations

Dataset	ϵ	PPR	HKPR	IPR	MEAN	GPR
DBLP	0.000000	0.1104	0.1125	0.1307	0.0958‡	**0.0380***
	0.000001	0.1755	0.1791	0.2033	0.1454‡	**0.0918***
	0.000010	0.2599	0.2656	0.2457	0.2376‡	**0.1080***
	0.000100	0.2612	0.2665	0.2461	0.2410‡	**0.1386***
Amazon	0.000000	0.0566	0.0561	0.0828	0.0494‡	**0.0159***
	0.000001	0.1017	0.0984	0.1405	0.0780‡	**0.0605***
	0.000010	0.1497	0.1450	0.1616	0.1179‡	**0.0709***
	0.000100	0.1869	0.1831	0.1820	0.1647‡	**0.0815***
PubMed	0.000000	0.0830	0.0861	0.0990	0.0681‡	**0.0391***
	0.000001	0.0840	0.0870	0.1023	0.0685‡	**0.0388***
	0.000010	0.1343	0.1385	0.1797	0.1028‡	**0.0825***
	0.000100	0.2397	0.2498	0.2188‡	0.2542	**0.2082***
CiteSeer	0.000000	0.0363	0.0358	0.0482	0.0280‡	**0.0187***
	0.000001	0.0363	0.0359	0.0488	0.0280‡	**0.0184***
	0.000010	0.0380	0.0371	0.0562	0.0288‡	**0.0236***
	0.000100	0.0718	0.0697	0.0805	0.0496‡	**0.0343***
Cora	0.000000	0.0812	0.0792	0.0878	0.0685‡	**0.0305***
	0.000001	0.0813	0.0794	0.0878	0.0685‡	**0.0312***
	0.000010	0.0837	0.0812	0.0891	0.0685‡	**0.0378***
	0.000100	0.0929	0.0905	0.0964	0.0733‡	**0.0592***

‡ The values with ‡ are the best among baselines

* The values with * and in bold are the best results

the node randomly selected within the same cluster as the training seed nodes. It's represented by the *c* columns in Table 3. 2)In-Graph: We do inference on the node randomly selected from the whole graph. It's represented by the *g* columns in Table 3. We report the average `conductance` of the 50 testing nodes with the final model in each dataset.

Results with approximation We report the results of different datasets in turn with both in-cluster and in-graph situations, which have not been shown in Section 4 with Figure 3.

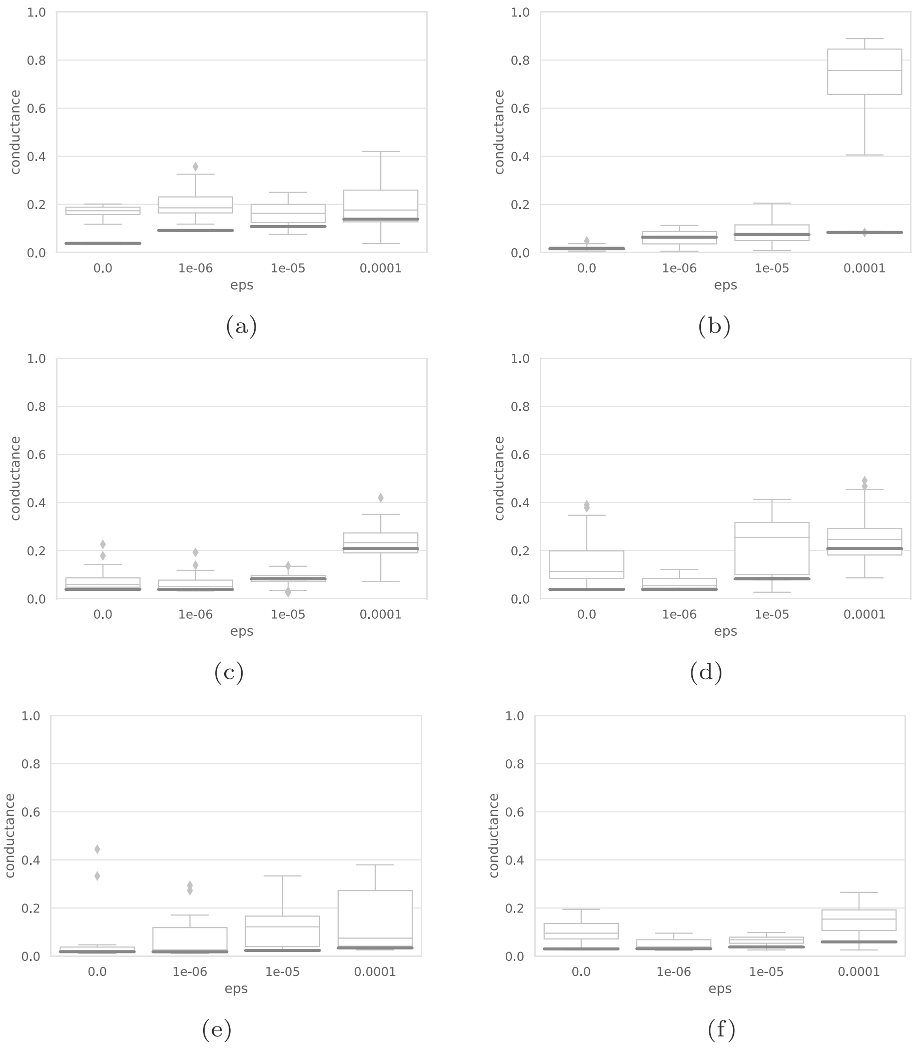

Figure 3 Generalization ability with approximation. (**a**) DBLP (**b**) Amazon (**c**) PubMed (**d**) PubMed* (**e**) CiteSeer (**f**) Cora*

A.4 Parameter sensitivity

Initialization comparisons We test the sensitivity of different initializations by training our LearnedNibble framework from different starting weights. Specifically, we use the PPR weighting vector with teleport constant $\alpha = 0.1$ to challenge our model. We use the IPR weighting vector with $\theta = 0.99, \phi = 0.99^{10}$ for IPR testing. The comparison results of different datasets are listed in Table 6. We can see that the training with different initialization methods achieves similar but slightly different performances. The trivial MEAN and RAW initializations perform a little better, and the IPR with theoretical advantage also plays well in some cases.

Regradient and locality regularization We investigate the regradient technique proposed in this work by conducting the ablation experiments. At the same time, we test the performance of the popular locality regularization term used in Graph Neural Networks(GNN), which keeps the information diffusion local with the minimizing the 2-norm of the difference between the graph signal after propagating and the initial signal which is the one-hot vector in our situation, i.e., $\|\mathbf{gpr} - \vec{1}_s\|$. The results under the exact settings with $\epsilon = 0$ of both are presented by Table 7. We can see that the regradient sets with $R = 1$ shows better performance than its comparisons with $R = 0$, and the training settings with $R = 1$; $L = 0$ corresponding to the experiments with regradient technique and without the commonly-used locality regularization achieves the best performance in all situations.

Table 6 Initialization sensitivity

Dataset	ϵ	PPR	IPR	MEAN	RAW
DBLP	0.000000	0.0431	0.0456	**0.0351***	0.0380
	0.000001	0.1015	0.1034	**0.0904***	0.0918
	0.000010	0.1368	0.1344	**0.1023***	0.1080
	0.000100	0.1709	0.1738	**0.1386***	**0.1386***
Amazon	0.000000	0.0160	**0.0153***	0.0170	0.0159
	0.000001	0.0745	0.0680	0.0661	**0.0605***
	0.000010	0.0825	0.0772	0.0761	**0.0709***
	0.000100	0.0989	0.0869	0.0934	**0.0815***
PubMed	0.000000	0.0386	0.0398	**0.0343***	0.0391
	0.000001	0.0388	0.0401	**0.0362***	0.0388
	0.000010	0.0822	**0.0777***	0.0839	0.0825
	0.000100	0.2201	0.2265	0.2083	**0.2082***
CiteSeer	0.000000	0.0217	0.0183	**0.0168***	0.0187
	0.000001	0.0212	0.0194	0.0189	**0.0184***
	0.000010	0.0242	0.0241	**0.0220***	0.0236
	0.000100	0.0375	0.0354	0.0349	**0.0343***
Cora	0.000000	0.0343	0.0325	0.0407	**0.0305***
	0.000001	0.0329	0.0323	0.0430	**0.0312***
	0.000010	0.0410	**0.0377***	0.0449	0.0378
	0.000100	0.0619	0.0601	0.0594	**0.0592***

* The values with * and in bold are the best results

Table 7 Ablation experiments

Dataset	R = 0;L = 1	R = 0;L = 0	R = 1;L = 1	R = 1;L = 0
DBLP	0.0494	0.0488	0.0431	**0.0364**[*]
Amazon	0.0160	0.0210	0.0159	**0.0151**[*]
PubMed	0.0399	0.0333	0.0503	**0.0330**[*]
CiteSeer	0.0262	0.0204	0.0187	**0.0182**[*]
Cora	0.0363	0.0344	0.0336	**0.0305**[*]

R = 0;L = 1 means without `regradient` but with locality regularization

R = 0;L = 0 means without `regradient` and without locality regularization

R = 1;L = 1 means with `regradient` and with locality regularization

R = 1;L = 0 means with `regradient` but without locality regularization

[*] The values with * and in bold are the best results

Acknowledgements The author would like to thank Wang Hanzhi and Zhang Ruoqi for their selfless and solid technical supports. This work is partially supported by the Fundamental Research Funds for the Central Universities (No.2020JS005).

Author contributions Yuan Zhe devised the methods and framework, wrote the whole manuscript text and prepared all materials.

Funding This work is partially supported by the Fundamental Research Funds for the Central Universities (No.2020JS005).

Data availability The graph datasets that support the findings of this study are available in SNAP project, https://snap.stanford.edu/data/index.html.

Declarations

Human and Animal Ethics Not applicable.

Ethics approval and consent to participate Not applicable.

Consent for Publication Not applicable.

Competing interests The author have no relevant financial or non-financial interests to disclose.

References

1. Girvan, M., Newman, M.E.: Community structure in social and biological networks. Proc. Nat. Acad. Sci. **99**(12), 7821–7826 (2002)
2. Wasserman, S., Faust, K., et al.: Social network analysis: Methods and applications (1994)
3. Boccaletti, S., Latora, V., Moreno, Y., Chavez, M., Hwang, D.-U.: Complex networks: Structure and dynamics. physrep **424**(4–5), 175–308 (2006). https://doi.org/10.1016/j.physrep.2005.10.009
4. Lu, Z., Wahlström, J., Nehorai, A.: Community detection in complex networks via clique conductance. Sci. Rep. **8**(1), 1–16 (2018)
5. Wang, M., Wang, C., Yu, J.X., Zhang, J.: Community detection in social networks: an in-depth benchmarking study with a procedure-oriented framework. Proc. VLDB Endow. **8**(10), 998–1009 (2015)
6. Fortunato, S.: Community detection in graphs. Phys. Rep. **486** (3–5), 75–174 (2010)

7. Leskovec, J., Lang, K.J., Mahoney, M.: Empirical comparison of algorithms for network community detection. In: Proceedings of the 19th International Conference on World Wide Web, pp. 631–640 (2010)
8. Yi, F., Moon, I.: Image segmentation: A survey of graph-cut methods. In: 2012 International Conference on Systems and Informatics (ICSAI2012), pp. 1936–1941. IEEE (2012)
9. Vicente, S., Kolmogorov, V., Rother, C.: Graph cut based image segmentation with connectivity priors. In: 2008 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8. IEEE (2008)
10. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. Int. J. Comput. Vis. **59**(2), 167–181 (2004)
11. Tolliver, D.A., Miller, G.L.: Graph partitioning by spectral rounding: Applications in image segmentation and clustering. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), vol. 1, pp. 1053–1060. IEEE (2006)
12. Liao, C. -S., Lu, K., Baym, M., Singh, R., Berger, B.: Isorankn: Spectral methods for global alignment of multiple protein networks. Bioinformatics **25**(12), 253–258 (2009)
13. Voevodski, K., Teng, S. -H., Xia, Y.: Finding local communities in protein networks. BMC Bioinform. **10**(1), 1–14 (2009)
14. Zhou, S., Yang, X., Chang, Q.: Spatial clustering analysis of green economy based on knowledge graph. Journal of Intelligent & Fuzzy Systems (Preprint), 1–10 (2021)
15. Foysal, K.H., Chang, H.J., Bruess, F., Chong, J.W.: Smartfit: Smartphone application for garment fit detection. Electronics **10**(1), 97 (2021)
16. Zhu, D., Shen, G., Chen, J., Zhou, W., Kong, X.: A higher-order motif-based spatiotemporal graph imputation approach for transportation networks. Wirel. Commun. Mob. Comput., 2022 (2022)
17. Spielman, D.A., Teng, S. -H.: Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In: Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing, pp. 81–90 (2004)
18. Andersen, R., Chung, F., Lang, K.: Local graph partitioning using pagerank vectors. In: 2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06), pp. 475–486. IEEE (2006)
19. Andersen, R., Peres, Y.: Finding sparse cuts locally using evolving sets. In: Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing, pp. 235–244 (2009)
20. Spielman, D.A., Teng, S. -H.: A local clustering algorithm for massive graphs and its application to nearly linear time graph partitioning. SIAM J. Comput. **42**(1), 1–26 (2013)
21. Lovász, L., Simonovits, M.: The mixing rate of markov chains, an isoperimetric inequality, and computing the volume. In: Proceedings [1990] 31st Annual Symposium on Foundations of Computer Science, pp. 346–354. IEEE (1990)
22. Lovász, L., Simonovits, M.: Random walks in a convex body and an improved volume algorithm. Random Struct. Algor. **4**(4), 359–412 (1993)
23. Andersen, R., Chung, F.: Detecting sharp drops in pagerank and a simplified local partitioning algorithm. In: International Conference on Theory and Applications of Models of Computation, pp. 1–12. Springer (2007)
24. Chung, F.: The heat kernel as the pagerank of a graph. Proc. Natl. Acad. Sci. **104**(50), 19735–19740 (2007)
25. Kloster, K., Gleich, D.F.: Heat kernel based community detection. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1386–1395 (2014)
26. Li, P., Chien, I., Milenkovic, O.: Optimizing generalized pagerank methods for seed-expansion community detection. Adv. Neural Inf. Process. Syst., 32 (2019)
27. Wang, H., He, M., Wei, Z., Wang, S., Yuan, Y., Du, X., Wen, J.-R.: Approximate graph propagation. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 1686–1696 (2021)
28. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the Web. Stanford InfoLab, Technical report (1999)
29. Chung, F., Simpson, O.: Solving linear systems with boundary conditions using heat kernel pagerank. In: International Workshop on Algorithms and Models for the Web-Graph, pp. 203–219. Springer (2013)
30. Yang, R., Xiao, X., Wei, Z., Bhowmick, S.S., Zhao, J., Li, R. -H.: Efficient estimation of heat kernel pagerank for local clustering. In: Proceedings of the 2019 International Conference on Management of Data, pp. 1339–1356 (2019)
31. Flake, G.W., Lawrence, S., Giles, C.L.: Efficient identification of web communities. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 150–160 (2000)
32. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **22**(8), 888–905 (2000)
33. Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., Parisi, D.: Defining and identifying communities in networks. Proc. Nat. Acad. Sci. **101**(9), 2658–2663 (2004)

34. Newman, M.E.: Modularity and community structure in networks. Proc. Nat. Acad. Sci. **103**(23), 8577–8582 (2006)
35. Kobourov, S.G., Pupyrev, S., Simonetto, P.: Visualizing graphs as maps with contiguous regions. In: EuroVis (Short Papers) (2014)
36. Cheeger, J.: A lower bound for the smallest eigenvalue of the Laplacian. Probl. Anal. **625**(195-199), 110 (1970)
37. Cox, I.J., Rao, S.B., Zhong, Y.: “ratio regions”: a technique for image segmentation. In: Proceedings of 13th International Conference on Pattern Recognition, vol. 2, pp. 557–564. IEEE (1996)
38. Sharon, E., Galun, M., Sharon, D., Basri, R., Brandt, A.: Hierarchy and adaptivity in segmenting visual scenes. Nature **442**(7104), 810–813 (2006)
39. Yang, J., Leskovec, J.: Defining and evaluating network communities based on ground-truth. Knowl. Inf. Syst. **42**(1), 181–213 (2015)
40. Benson, A.R., Gleich, D.F., Leskovec, J.: Higher-order organization of complex networks. Science **353**(6295), 163–166 (2016)
41. Tsourakakis, C.E., Pachocki, J., Mitzenmacher, M.: Scalable motif-aware graph clustering. In: Proceedings of the 26th International Conference on World Wide Web, pp. 1451–1460 (2017)
42. Yin, H., Benson, A.R., Leskovec, J., Gleich, D.F.: Local higher-order graph clustering. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 555–564 (2017)
43. Ma, W., Cai, L., He, T., Chen, L., Cao, Z., Li, R.: Local expansion and optimization for higher-order graph clustering. IEEE Internet Things J. **6**(5), 8702–8713 (2019)
44. Huang, S., Li, Y., Bao, Z., Li, Z.: Towards efficient motif-based graph partitioning: An adaptive sampling approach. In: 2021 IEEE 37th International Conference on Data Engineering (ICDE), pp. 528–539. IEEE (2021)
45. Zhou, D., Zhang, S., Yildirim, M.Y., Alcorn, S., Tong, H., Davulcu, H., He, J.: High-order structure exploration on massive graphs: A local graph clustering perspective. ACM Trans. Knowl. Discov. Data (TKDD) **15**(2), 1–26 (2021)
46. Chhabra, A., Faraj, M.F., Schulz, C.: Local motif clustering via (hyper) graph partitioning. arXiv:2205.06176 (2022)
47. Emmons, S., Kobourov, S., Gallant, M., Börner, K.: Analysis of network clustering algorithms and cluster quality metrics at scale. PloS one **11**(7), 0159161 (2016)
48. Shannon, C.E.: A mathematical theory of communication. Bell Syst. Tech. J. **27**(3), 379–423 (1948)
49. Meilă, M.: Comparing clusterings—an information based distance. J. Multivar. Anal. **98**(5), 873–895 (2007)
50. Vinh, N.X., Epps, J., Bailey, J.: Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. J. Mach. Learn. Res. **11**, 2837–2854 (2010)
51. Avron, H., Horesh, L.: Community detection using time-dependent personalized pagerank. In: International Conference on Machine Learning, pp. 1795–1803. PMLR (2015)
52. Kloumann, I.M., Ugander, J., Kleinberg, J.: Block models and personalized pagerank. Proc. Natl. Acad. Sci. **114**(1), 33–38 (2017)
53. Li, Y., Liu, J., Lin, G., Hou, Y., Mou, M., Zhang, J.: Gumbel-softmax-based optimization: a simple general framework for optimization problems on graphs. Comput. Soc. Netw. **8**(1), 1–16 (2021)
54. Holland, P.W., Laskey, K.B., Leinhardt, S.: Stochastic blockmodels: First steps. Soc. Netw. **5**(2), 109–137 (1983)
55. Weiss, P.: L’hypothèse du champ moléculaire et la propriété ferromagnétique. J. Phys. Theor. Appl. **6**(1), 661–690 (1907)
56. Klicpera, J., Weißenberger, S., Günnemann, S.: Diffusion improves graph learning. Advances in Neural Information Processing Systems, 32 (2019)
57. Berberidis, D., Nikolakopoulos, A.N., Giannakis, G.B.: Adaptive diffusions for scalable learning over graphs. IEEE Trans. Signal Process. **67**(5), 1307–1321 (2018)
58. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv:1412.6980 (2014)
59. Leskovec, J., Sosič, R.: Snap: A general-purpose network analysis and graph-mining library. ACM Trans Intelli Syst Technol (TIST) **8**(1), 1–20 (2016)
60. Getoor, L.: Link-based classification. In: Advanced Methods for Knowledge Discovery from Complex Data, pp. 189–207. Springer (2005)
61. Namata, G., London, B., Getoor, L., Huang, B., EDU, U.: Query-driven active surveying for collective classification. In: 10th International Workshop on Mining and Learning with Graphs, vol. 8, p. 1 (2012)