Nonlinear Diffusion for Community Detection and Semi-Supervised Learning

Rania Ibrahim
Computer Science Department,
Purdue University
West Lafayette, Indiana, USA
ibrahimr@purdue.edu

David F. Gleich
Computer Science Department,
Purdue University
West Lafayette, Indiana, USA
dgleich@purdue.edu

ABSTRACT

Diffusions, such as the heat kernel diffusion and the PageRank vector, and their relatives are widely used graph mining primitives that have been successful in a variety of contexts including community detection and semi-supervised learning. The majority of existing methods and methodology involves linear diffusions, which then yield simple algorithms involving repeated matrix-vector operations. Recent work, however, has shown that sophisticated and complicated techniques based on network embeddings and neural networks can give empirical results superior to those based on linear diffusions. In this paper, we illustrate a class of nonlinear graph diffusions that are competitive with state of the art embedding techniques and outperform classic diffusions. Our new methods enjoy much of the simplicity underlying classic diffusion methods as well. Formally, they are based on nonlinear dynamical systems that can be realized with an implementation akin to applying a nonlinear function after each matrix-vector product in a classic diffusion. This framework also enables us to easily integrate results from multiple data representations in a principled fashion. Furthermore, we have some theoretical relationships that suggest choices of the nonlinear term. We demonstrate the benefits of these techniques on a variety of synthetic and real-world data.

CCS CONCEPTS

• **Mathematics of computing** → **Graph algorithms**; • **Computing methodologies** → *Cluster analysis*; • **Theory of computation** → *Semi-supervised learning*.

KEYWORDS

Nonlinear Diffusion; Community Detection; Graph-based Semi-Supervised Learning

ACM Reference Format:
Rania Ibrahim and David F. Gleich. 2019. Nonlinear Diffusion for Community Detection and Semi-Supervised Learning. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313483

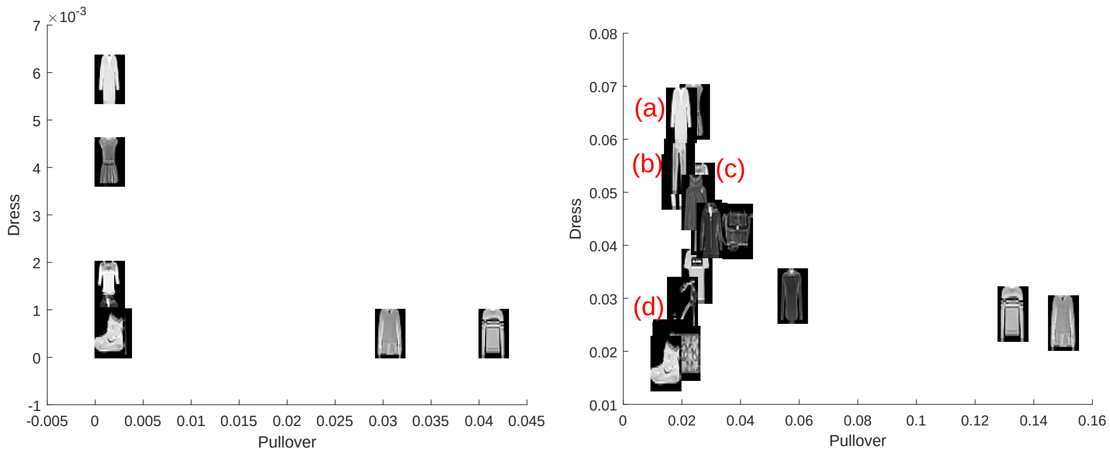
1 INTRODUCTION

Diffusion methods for graph mining work by modeling a process where we conceptually inject a *dye* at a particular node or set of nodes in the graph and observe how the *dye* diffused over the edges of the graph.[1] How the *dye* diffuses reveals latent structure in the graph. For instance, if the *dye* stays nearby the source, this indicates that there is a community nearby the source. If the *dye* starts at nodes with a particular label, then under homophily (where neighbors share similar properties), the *dye* will concentrate at nodes likely to share the same label. In this way, the method can be used for semi-supervised learning. Because this conceptual view of diffusions is general, a graph diffusion can be realized in a large number of ways including smooth functions on graphs [57], random walks [38], PageRank equations [37], and linear algebra relaxations of combinatorial graph theory [15, 35]. Consequently, diffusion methods have been widely applied to a large variety of graph mining tasks [13, 18, 20, 24, 32, 38]. In the majority of these results, diffusion methods gave a performance that was comparable to the state of the art, with many possible ways to adjust the computational primitive to improve the results and to scale up to massive problems.

More recently, however, a new class of techniques emerged that raised the state of the art performance on semi-supervised learning problems. These are based on node embeddings along with subsequent machine learning algorithms [22, 39] or from more complicated graph-based variations on neural networks [26, 56]. These methods are impressive because they have established new benchmarks for performance on learning with graph-based data. A downside to these methods, however, is that they can be hard to reason about and can make it difficult to understand what may be causing poor performance on new problems.

Diffusions, on the other hand, are highly intuitive, simple procedures modeled on the *dye* description in the introduction. The vast majority of existing work on diffusions in graphs considers linear diffusions, those where the dynamics can be fully captured by a single matrix operator, such as a random-walk operator or a Laplacian matrix, or a regularized or perturbed version of either [19]. In this paper, we consider nonlinear diffusions, which arise in more

[1]We use the term *dye* here as an intuitive concept only and mimic the ideas of dyes and diffusions in medical diagnostics and physics.



Linear heat diffusion *Our new nonlinear fast heat diffusion*

Figure 1: Comparison between linear diffusion based on the linear heat kernel diffusion (left) and our nonlinear diffusion (right) on Fashion MNIST dataset where we seeded the diffusions on a few pictures of pullover sweaters and dresses. This gives a coordinate for each image in the graph based on its diffusion value and these diffusion values reflect how much it looks like a pullover or dress. We draw the same 20 examples for linear diffusion and nonlinear diffusion. The linear diffusion concentrates the results nearby the seeds (most images are on top of each other) where the nonlinear diffusion captures our intuitive notion that there is a diversity of images that look like dresses or pullovers. For example, in the right figure, with respect to the y-axis, the sorting of the diffusion values reflect the semantic similarity between the clothes. Marker (a) points to dresses, marker (b) points to trousers (long as dress), marker (c) points to tops and finally, marker (d) points to shoes.

complicated physics and ecology settings. For instance, the porous media equation [23, 29] describes diffusion through a physical substance that can capture and store the diffusing substance (such as an isentropic gas), slowing the diffusion in a nonlinear fashion. The fast diffusion equation [48] describes diffusion in plasma's [6], and also the diffusion of materials in silicon when the medium is clustered [25]. These diffusions are fairly well understood from an abstract mathematical perspective (e.g. see [7, 48, 49]). With only a few exceptions, however, these ideas have not been applied to graph mining tasks. (See related work for more discussion of what has been done with respect to graphs and networks.) A related type of nonlinearity, which involves using a nonlinear transmission process which diffuses *dye* according to some nonlinear rule, is captured with a nonlinear Laplacian operator such as the p-Laplacian [1, 8, 9]. These have been studied as ways to generalize spectral clustering (see Section 4 for more about the differences). Additionally, it was shown in [36] that using the traditional Laplacian in semi-supervised learning results in labeling function that is spiky at the labeled points and flat everywhere else. Bridle et al. [8] solve this problem using the p-Laplacian in an energy minimization problem and we inherit this advantage when we use the p-Laplacian in the diffusion process.

In this paper, we show that by using values from these nonlinear diffusions, we are able to match state of the art performance benchmarks in semi-supervised learning. This helps to retain the intuitive reasoning behind the methods. As an example of how nonlinear diffusions help to solve problems, see Figure 1, where we show the solution of the standard heat kernel diffusion and nonlinear diffusion for a semi-supervised learning problem on the Fashion MNIST dataset [54]. In this case, we form a graph based on the images themselves (see more details in Section 6.4). Then we diffuse from a few images of a pull-over sweater and a dress with the heat kernel diffusion and our nonlinear diffusion. We show each image (node in the graph) in the after the diffusion has run for some time for the kernel diffusion compared with the nonlinear fast diffusion. As the figure shows, using the nonlinear diffusion causes the points to spread out far more than what is shown in the heat kernel. This illustrates how these advanced diffusions can give a much wider spread of values compared with standard diffusions and enable better classification. Avoiding these *spiked* solutions was also a motivation of past work on nonlinear diffusions [8].

More generally, introducing simple nonlinearities often improves the performance of other machine learning tasks. For example, nonlinear dimensionality reduction techniques [41, 46] outperforms linear techniques by assuming that the data lies in a nonlinear manifold. Nonlinear activation functions enhance the results of neural networks [16], and simple nonlinear transformations of the results of spectral clustering improve results as well [51]. Hence, our work shows yet another way that simple nonlinearities can improve the performance of graph mining tasks.

Finally, one of the great advantages to diffusion methods is that they offer theoretical guarantees of performance through Cheeger inequalities [11–13, 35, 55]. These relate the output from the diffusion to nearby bottlenecks in the graph, formalized as low conductance sets. (See [56] for examples on how general this perspective can be to capture a large number of notions of *conductance*.) In this paper, we have results that suggest using a nonlinear diffusion using the power function with power value close to 0 should give the best performance for low-conductance set detection through new analysis of the sweepcut procedure (Theorem 5.8).

Our experimental results (Section 6) show that nonlinear diffusion outperforms heat diffusion in community detection by up to 35% in F1-measure and it outperforms related work in graph-based semi-supervised learning in most datasets. In the latter case, we show how our diffusion framework can be adapted to work on multiple datasets (Section 3.2).

Summary of contributions and outline of the paper.

- We introduce two methods for nonlinear diffusion, one using nonlinear functions and another one using nonlinear Laplacian operators (Section 2).
- We place these diffusions in a framework where we can integrate multiple data representations and to self-train itself (Section 3).
- We show that nonlinear diffusion with power function should give smaller conductance sets with p approaches 0. (Section 5).
- We experimentally compare nonlinear diffusion methods to related work in community detection and graph-based semi-supervised learning (Section 6).

2 EVALUATING NONLINEAR DIFFUSIONS

2.1 Notation and Background

We use the following notation throughout the rest of the paper. Scalars are denoted by small letters (e.g., m, n), sets are shown in capital letters (e.g., X, Y), vectors are denoted by small bold letters (e.g., $\mathbf{f}, \mathbf{g}$), and matrices are denoted by capital bold letters (e.g., $\boldsymbol{A}, \boldsymbol{L}$).

Let $G = (V, E)$ be an undirected graph, where $n = |V|$ is the number of nodes in the graph. Let $\boldsymbol{A}$ be the associated adjacency matrix of the graph G and $\boldsymbol{D}$ be the diagonal matrix of degrees, where $\boldsymbol{D}_{ii} = \mathbf{d}(i)$, is the degree of the vertex i. (In the case of a weighted graph, then the weighted degrees are the sum of the edge weights.) Additionally, $\mathbf{e}_S$ is the vector of all ones for entries $i \in S$ and zero otherwise.

Conductance. Conductance is one of the popular quality measures that quantifies how good is the set S as a partition of the graph G [43]. Given a subset S of the vertices V, we define the volume of set S as $\text{vol}(S) = \sum_{v \in S} \mathbf{d}(v)$ and the cut of set S as $\text{cut}(S) = \{\{u, v\} \in E : u \in S \text{ and } v \in \bar{S}\}$. Furthermore, the conductance of the set S is defined as:

$$\phi(S) = \frac{|\text{cut}(S)|}{\text{minvol}(S)},$$

where $\text{minvol}(S) = \min(\text{vol}(S), \text{vol}(V - S))$. For weighted graphs, the volume uses the weighted degrees and the $|\text{cut}(S)|$ is the sum of edge weights cut.

The heat kernel diffusion The heat kernel diffusion [13] from a seed set S results in a solution to the dynamical system:

$$\frac{\partial \mathbf{u}}{\partial t} = -\boldsymbol{L}\mathbf{u} \qquad \mathbf{u}(0) = \frac{\mathbf{e}_S}{|S|},$$

where $\boldsymbol{L} = \boldsymbol{I} - \boldsymbol{A}\boldsymbol{D}^{-1}$ is the random walk Laplacian with eigenvalues $\lambda \in [0, 2]$.[2]

[2] Note that the Fiedler vector used in spectral clustering is $\mathbf{f} = \boldsymbol{D}\mathbf{x}_2$, where $\mathbf{x}_2$ is the eigenvector associated with the second smallest eigenvalue of $\boldsymbol{L}$.

We can state a closed-form solution via the matrix exponential:

$$\mathbf{u}(t) = \exp(-t\boldsymbol{L})\mathbf{e}_S/|S|,$$

which has been shown to have a number of nice properties [13, 27].

In the remainder of this section, we will describe two methods to encode nonlinearity in the heat kernel diffusion. The first method is by first applying a nonlinear function g to grow or decay the values at each node and then transmitting the heat according to the heat diffusion differential equation. The second method is by using a nonlinear Laplacian operator that grows or decays the amount transferred on each edge.

2.2 Nonlinear Diffusion with Growth or Decay

The following dynamical system describes a general notion of nonlinear diffusion (e.g. [49]) when applied to a graph or network:

$$\frac{\partial \mathbf{u}}{\partial t} = -\boldsymbol{L}g(\mathbf{u}) \qquad \mathbf{u}(0) = \frac{\mathbf{e}_S}{|S|}.$$

Here g is the nonlinear function, S is the set of seed nodes, and $\mathbf{u}(t) \in \mathbb{R}^n$ is the diffusion value at each node at time t. One representative choice of g is the power function $g(\mathbf{u}) = \mathbf{u}^p$ (element-wise). Using the power function is motivated by the porous media equation ($p > 1$) [47] and the fast diffusion equation ($p < 1$) [7] (see discussion of these in the introduction). Setting $p = 1$ gives the linear heat kernel diffusion [13].

In nonlinear diffusion, each node will apply the function g to its current diffusion value before transmitting the heat to its neighbor. If the values of $\mathbf{u} \in [0, 1]$, then if $p > 1$, the diffusion at each node is reduced first according to both the parameter p and the diffusion value at the node before the node can transmit any heat. This models the diffusion medium *capturing* the value being diffused. Hence, this will keep the diffusion more localized around the seed nodes. However, if $p < 1$, then the diffusion at each node will be magnified first before the node starts transmitting the value, which will help the diffusion explore wider regions. This models settings where large values diffuse faster.

Due to the effect of the nonlinearity, nonlinear diffusion does not have a closed form solution as the heat kernel diffusion, therefore we approximate the solution of the dynamical system by using a simple forward Euler integration, which can be derived by approximating the time-derivative $\partial\mathbf{u}/\partial t \approx (\mathbf{u}(t + h) - \mathbf{u}(t))/h$.[3] Hence, we have the evolution:

$$\mathbf{u}(t + h) = \mathbf{u}(t) - h\boldsymbol{L}\mathbf{u}(t)^p.$$

Here $\mathbf{u}(t + h)$ is the diffusion value at time $t + h$. Because the notion of "time" for this system is arbitrary, we only study the solution after k steps of the evolution process with a fixed value of h. The solution $\mathbf{u}$ for the dynamical systems is non-negative as mentioned in [4] and [47]. However, as we approximate the dynamical system using forward Euler method, the values can be negative. Getting non-negative numerical solutions is difficult as discussed in [44] and therefore, to ensure that $\mathbf{u}$ values are between 0 and 1, we truncate values less than zero to be zero and values greater than one to be one. Algorithm 1 gives a simple pseudocode to do this.

[3] The forward Euler method is often problematic in dynamical systems in physics because there are conservation laws that need to be respected in the approximation. Because the diffusion framework here is an abstraction, we are less worried about accuracy in terms of the ODE, which makes forward Euler a simple, pragmatic choice.

We have ongoing analysis to study accuracy and convergence in the integration problem in more detail that are beyond the scope of this paper; among these results, our analysis gives a direct proof of non-negativity.

Algorithm 1: Nonlinear Diffusion with Growth or Decay

```
1 NonlinearDiffusionWithGrowthOrDecay ($L, k, h, g, \mathbf{s}$)
  Input  : Random walk Laplacian $L$, number of steps $k$, step
           length $h$, nonlinear function $g$ and initial diffusion
           values $\mathbf{s}$.
  Output : $\mathbf{f}$
2 $\mathbf{u} = \mathbf{s}$ ;
3 for $i \leftarrow 0$ to $k$ do
4     $\mathbf{u} = \mathbf{u} - hLg(\mathbf{u})$ ;
5     truncate values of $\mathbf{u}$ to be between 0 and 1
6 end
7 $\mathbf{f} = g(\mathbf{u})$;
```

2.3 Nonlinear Diffusion via Nonlinear Transfer

Like the heat kernel diffusion, we can also define nonlinear diffusion using nonlinear transfer. One way to do this is via the p-Laplacian [1, 8, 9]. The p-Laplacian operator [8, 9] is defined as:

$$(\nabla_p \mathbf{f})_i = \sum_{j \in V} w_{ij} \phi_p(\mathbf{f}_i - \mathbf{f}_j),$$

where $\phi_p : \mathbb{R} \to \mathbb{R}$ is

$$\phi_p(x) = |x|^{p-1} \text{sign}(x),$$

and $\mathbf{f} = D^{-1}\mathbf{u}$. This results in a nonlinear Laplacian operator as in the following setting:

$$\frac{\partial \mathbf{u}}{\partial t} = \nabla_p \mathbf{u}.$$

We can write the result of the p-Laplacian on a vector $\mathbf{u}$ using the incidence matrix of the graph. Let N be the edges-by-nodes, unweighted incidence matrix and W be a diagonal matrix of edge-weights that corresponds to the order of edges in the incidence matrix.[4] Then

$$\phi_p(\mathbf{f}_i - \mathbf{f}_j) = |ND^{-1}\mathbf{u}|^{p-1} .* \text{sign}(ND^{-1}\mathbf{u})$$
$$\nabla_p \mathbf{f} = N^T W |ND^{-1}\mathbf{u}|^{p-1} .* \text{sign}(ND^{-1}\mathbf{u}).$$

Here, we use the element-wise multiplication .*, absolute value and power. Additionally, we use the observation that $\phi_p(ND^{-1}\mathbf{u})$ has the same sign as $ND^{-1}\mathbf{u}$. Hence when we multiply by $N^T W$ the arbitrary sign of the difference of edges in the incidence matrix cancels and we scale by the appropriate weight because of W. Note that if $p = 2$, then the equation corresponds to the heat kernel diffusion. Algorithm 2 shows the pseudocode for the nonlinear diffusion using the p-Laplacian.

[4]For the unweighted incidence matrix N and an unweighted graph, $L = I - AD^{-1} = N^T N D^{-1}$. For a weighted graph, we have $L = N^T W N D^{-1}$. The matrices N and W are easy to build from the raw list of edges. If `ei`, `ej`, `ew` are length $|E| = m$ arrays with the source, destination, and weight of each edge, then N = `sparse([1 : m; 1 : m], [ei; ej], [ones(m, 1); −ones(m, 1)], m, n)` and W = `diag(ew)` in pseudo-Matlab or Julia notation.

Algorithm 2: Nonlinear Diffusion using the p-Laplacian

```
1 NonlinearTransfer ($N, \mathbf{w}, D, k, h, p, \mathbf{s}$)
  Input  : Unweighted incidence $N$, edge weights $\mathbf{w}$, degree
           matrix $D$, number of steps $k$, step length $h$, nonlinear
           diffusion parameter $p$ and initial diffusion values $\mathbf{s}$.
  Output : $\mathbf{f}$
2 $\mathbf{u} = \mathbf{s}$ ;
3 for $i \leftarrow 0$ to $k$ do
4     $\mathbf{q} = ND^{-1}\mathbf{u}$ ;
5     $\mathbf{u} = \mathbf{u} - hN^T[\mathbf{w} .* (|\mathbf{q}|)^{p-1} .* \text{sign}(\mathbf{q})]$ ;
6     truncate values of $\mathbf{u}$ to be between 0 and 1
7 end
8 $\mathbf{f} = D^{-1}\mathbf{u}$;
```

3 USING NONLINEAR DIFFUSIONS

Given an input graph G, we can use the solution vector $\mathbf{f}$ that results from either of the two nonlinear diffusions instead of any of the vectors used in semi-supervised learning [19, 24, 35, 57] or local community detection [13, 27, 55] as a straightforward substitution in these algorithmic pipelines. All of these pipelines feature a *diffuse-and-round* perspective where the results of multiple diffusions are turned into estimated classes in the case of semi-supervised learning or rounded to clusters in the case of community detection. We elaborate more on community detection in Section 5.

To be more precise, for semi-supervised learning, we assume we have $S_1, \ldots, S_c$ as c-seed sets that represent samples of c classes. The goal of the problem is to infer the class label $1, \ldots, c$ for the remaining nodes $V - \cup S_i$. The most interesting case is when $|S_i|$ is small and constant, such as 5 or 10 and independent of the graph size. In which case, we diffuse from S_i using our nonlinear diffusion to compute a vector $\mathbf{f}_i$. With all the vectors $\mathbf{f}_1, \ldots, \mathbf{f}_c$, we assign classes based on their rank in the diffused solutions [19]. That is, node j is the $r_1, \ldots, r_c$th largest node in $\mathbf{f}_1, \ldots, \mathbf{f}_c$, then we assign its class to $\text{argmin}_i\, r_i$. We can also pick the class based on the largest value of node j in any of the diffusions $[\mathbf{f}_i]_j$, which we call value-based rounding [57].

Complexity. The runtime complexity of our diffusions is linear in the number of edges of the graph, because the major piece of computation involves k-matrix vector products – multiplying by a Laplacian or incidence matrix is linear in the number of edges – for each of the c classes.

For the case of semi-supervised learning, our experiments and final methodology incorporate two additional ideas. First, we can use feedback from the diffusion process itself to suggest new training labels [10] (Section 3.1). Second, we show how to easily incorporate multiple data representations into our diffusion framework (Section 3.2).

3.1 Self-Training with Diffusion Methods

In semi-supervised learning, a self-trained diffusion means that the diffusion method incorporates feedback from its results. For instance, if the results of a diffusion $\mathbf{f}_i$ suggest that a subset of node R is extremely like to have label i, then we would re-run the diffusion using seeds $S_i \cup R$. This process is repeated either for a specific number of iterations or until no significant change is observed in the diffusion method. A confidence score for each label can be calculated based on the diffusion values. The higher the diffusion value, the more confident the diffusion is about its decision. As self-training depends on the decision of the diffusion, if the diffusion yields useful results, then the quality of the diffusion will enhance over time, otherwise, the quality will decrease. Therefore, there is no guarantee that self-training will always enhance the quality of the diffusion, however, in practice it often does.

3.2 Integrating multiple data representations

In many real-world learning problems, the underlying data has multiple representations. For example in citation graphs, there is a matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ which captures the relationship between publications based on citations but there are also metadata associated with each paper including authors, titles, and keywords. It is essential to make use of all the available representations in order to develop a better understanding of the data. Consequently, we show how to use 2 different types of relationships between our objects to do prediction. The extension to even more types is simple.

The idea is to construct multiple graphs based on each data representation. For example in citation graphs, we would have $\boldsymbol{A}$ the citation graph, and also $\boldsymbol{B}$, which is a nearest neighbor graph based on some similarity function over the metadata features. Then, we want to run diffusions in $\boldsymbol{A}$ and $\boldsymbol{B}$ simultaneously and our goal is to have all the diffusions concentrated in the same set of nodes. In order to accomplish this goal, each diffusion will try to pull the other diffusions to be similar to it. This changes the nonlinear diffusion differential equation (and similarly the heat kernel diffusion) to:[5]

$$\mathbf{u}_{k+1}^{A} = \mathbf{u}_{k}^{A} - h\boldsymbol{L}_{A}g(\mathbf{u}_{k}^{A}) - \sigma(\mathbf{u}_{k}^{A} - \mathbf{u}_{k}^{B})$$
$$\mathbf{u}_{k+1}^{B} = \mathbf{u}_{k}^{B} - h\boldsymbol{L}_{B}g(\mathbf{u}_{k}^{B}) - \sigma(\mathbf{u}_{k}^{B} - \mathbf{u}_{k}^{A}).$$

Here $\mathbf{u}_k^A$ is the diffusion value for the diffusion running over $\boldsymbol{A}$ at step k and we describe the update to get to the next step. The impact of the parameter σ is the strength of the coupling between the solutions. If σ is large, then we will move the solutions together. If σ is small, then they will evolve with only weak coupling.

4 RELATED WORK

Diffusion methods for clustering, semi-supervised learning, and a variety of other problems have a lengthy history, which has been extensively discussed in various surveys [10, 18, 34]. These methods are all based on linear notions of diffusion. We highlight two important connections we use in the remainder of the paper: spectral clustering and personalized PageRank. For a useful broad perspective on diffusion, we also recommend Ghosh et al. and Yan et al. [17, 55].

Spectral clustering is based on looking for minimum energy modes of the graph Laplacian. These minimum energy modes also can be interpreted as bottlenecks in a diffusion over the nodes. Since this is a linear matrix operator, these minimum energy modes correspond to the small eigenvectors. Specifically, the Fiedler vector is the second smallest generalized eigenvector $\boldsymbol{LD}\mathbf{f} = \lambda \boldsymbol{D}\mathbf{f}$. This eigenvector gives the smallest non-trivial minimizer of the energy function $E(\mathbf{f}) = \mathbf{f}^T \boldsymbol{LD}\mathbf{f}/(\mathbf{f}^T\boldsymbol{D}\mathbf{f})$. The Fiedler vector provably identifies a set of small conductance in the graph [15].

Personalized PageRank (PPR) is a diffusion method that models a random walk with reset process [37]. As a diffusion, it corresponds to the infinite time limit of

$$\frac{d\mathbf{x}}{dt} = (1-\alpha)(\mathbf{e}_S - \mathbf{x}) - \alpha \boldsymbol{L}\mathbf{x},$$

following results on the dynamics of PageRank [21]. PageRank can also be derived as a locally biased analogue of spectral clustering [19], and it is commonly used both for semi-supervised learning [57] and community detection [2], where it also identifies a set of small conductance in the graph. We compare against PPR in our experiments.

4.1 Nonlinear diffusions in graphs.

All of the existing work on nonlinear diffusion for graphs uses the p-Laplacian [1] and seeks to generalize spectral clustering inspired approaches. For instance, Buhler and Hein [9] seek to generalize the Fiedler vector to the p-Laplacian by finding low-energy states. Likewise, Brindel and Zhu [8] seek low-energy states of the p-Laplacian that fix the value of specific nodes to use for semi-supervised learning. Our perspective is different and we seek to generalize methods based on their diffusion interpretations rather than the energy minimization perspectives. For linear diffusions, the two are essentially equivalent because the eigenvectors determine the long-term behavior of the diffusion, but the generalizations are rather different for nonlinear operations.

4.2 Relationship to Neural Networks

Our work can also be interpreted in the context of graph-based neural networks [42]. These view a graph, or a matrix derived from a graph, as defining a layer in a neural network. Hence, we can interpret our diffusion as the output of a graph neural network with a fixed set of weights and a variable number of layers (depending on k). However, in addition to the graph structure, there are other sources of information that are not utilized in these related work like features per node. To use both the graph structure along with the feature matrix, both Planetoid [56] and Graph Convolutional Networks (GCN) [26] were proposed to allow neural networks to jointly learn from the feature matrix along with the graph structure. We compare against both of these methods in our experiments. (In this work, we extend the diffusion methods to learn from different sources of information.)

Compared to neural networks, nonlinear diffusion has two advantages, the first one is that it is more interpretable, as it follows a dynamical equation. Having a dynamical equation makes it easier to reason about its decisions and to develop theoretical results like developing Cheeger-type inequalities. The second advantage is that it reduces the number of parameters (we do no optimization over the weights of the layers) used in neural networks and therefore it is a simpler model.

4.3 Further relationships

Another line of work studies *fractional Laplacians*, which model the diffusion dynamics with fractional powers of the Laplacian matrix [40]. As such, these are still linear diffusions, albeit on a

[5]The equations for the p-Laplacian formulation are changed in an analogous fashion.

non-linear transformation of the Laplacian matrix. Our approach to integrating the results of multiple representations bears some relationship with recent work on centrality in temporal networks [45], although the motivation and applications are distinct. Finally, there are methods that use flow techniques for semi-supervised learning [50, 52]. These techniques offer fundamentally different tradeoffs. In our view, they are often best used to refine the results of diffusion algorithms to achieve better downstream performance.

5 NONLINEAR DIFFUSION ANALYSIS

The goal of this section is to prove our main theoretical result, which guides how we would pick a nonlinear diffusion to find small conductance sets in a graph. The precise method we study is as follows. We use Algorithm 1 to diffuse from a seed set S to produce a diffusion vector $\mathbf{f}$. (Note that $\mathbf{f}$ depends on the seed set, the value h, and the number of steps k.) Then we perform a sweep-cut procedure over the vector $\mathbf{f}$ to identify a set of small conductance. The sweep-cut procedure works by building a permutation $\sigma = (\sigma_1, \sigma_2, \dots, \sigma_n)$ that sorts $\mathbf{f}$ from large elements to small elements, and then evaluates the conductance of each sorted prefix $S_i = (\sigma_1, \dots, \sigma_i)$. We return the set $S = \operatorname{argmin}_i \phi(S_i)$ where $\phi(S_i)$ is the conductance.

This proof of our main results proceeds in two stages. First, we show a relationship between the sweep over any vector $\mathbf{f}$ and the set of minimal conductance (Section 5.2) that shows how well an "approximate indicator" vector behavior. Second, we will use the properties of nonlinear diffusion to guide our choice for nonlinear diffusion.

5.1 f-weighted Conductance

Before we begin, we fix a variety of notation that we will use throughout the proof and introduce a specialized notion of conductance, that we term f-weighted conductance. Our objective is to develop an inequality that relates the smallest conductance of any set, ϕ^*, in the graph G to the conductance of performing sweepcut over the vector $\mathbf{f}$, which results in conductance $\phi(\mathbf{f})$. Additionally, we want this inequality to maintain a linear relationship between ϕ^* and $\phi(\mathbf{f})$. This will require quantifying how well $\mathbf{f}$ looks like an indicator vector over the optimal set.

Using $\mathbf{f}$, we will direct each undirected edge $e = (u, v)$ in the graph G, such that it points from the node with higher $\mathbf{f}$ value to the node with lower $\mathbf{f}$ value. This re-direction will encode the sorted order of the nodes and will ensure that $\mathbf{f}(u) - \mathbf{f}(v) \geq 0$ for all $(u, v) \in E$. To encode this re-direction using matrices, we will define $\boldsymbol{N}_\mathbf{f} \in \{0, 1, -1\}^{m \times n}$ to be the $\mathbf{f}$-oriented incidence matrix, where for each edge $e = (u, v)$, $\boldsymbol{N}_\mathbf{f}(e, u) = 1$ and $\boldsymbol{N}_\mathbf{f}(e, v) = -1$ if and only if $\mathbf{f}(u) \geq \mathbf{f}(v)$. We will use $x \sim y$ to iterate over the directed edges. Additionally, assume that the vertices are numbered descending according to $\mathbf{f}$, so that we have $\mathbf{f}(v_1) \geq \mathbf{f}(v_2) \geq \dots \geq \mathbf{f}(v_n)$ and let us first normalize $\mathbf{f}$ so that its values are between 0 and 1.

To accomplish our objective, we need to define a relationship between the set S^* of smallest conductance in the graph and the set obtained by performing a sweepcut over $\mathbf{f}$. Therefore, we define the f-weighted conductance $\phi^*(\mathbf{f})$ as:

$$\phi^*(\mathbf{f}) = \frac{|\text{cut}_\mathbf{f}(S^*)|}{\text{minvol}_\mathbf{f}(S^*)},$$

where $|\text{cut}_\mathbf{f}(S^*)| = (\mathbf{e}^T_{S^* \to \bar{S^*}} + \mathbf{e}^T_{\bar{S^*} \to S^*})\boldsymbol{N}_\mathbf{f}\mathbf{f}$, $\mathbf{e}_{S \to \bar{S}}$ is the indicator vector where only edges pointing from S to $\bar{S}$ have value one and other edges have value zero, and $\text{minvol}_\mathbf{f}(S^*) = \min(\sum_{u \in S^*} \mathbf{f}(u)\mathbf{d}(u), \text{vol}(G) - \sum_{u \in S^*} \mathbf{f}(u)\mathbf{d}(u))$. If $\mathbf{f} = \mathbf{e}_{S^*}$, then $\phi^*(\mathbf{f}) = \phi^*$.

5.2 A General Sweepcut Inequality

To derive a general sweepcut inequality, we will derive an upper bound and a lower bound for $\phi^*(\mathbf{f})$ and using these bounds, we will be able to relate $\phi(\mathbf{f})$ to ϕ^* linearly. Additionally, let us define W as the set of nodes that have nonzero $\mathbf{f}$ values.

Lemma 5.1. *If $vol(W) \leq minvol(S^*)$, then:*

$$\phi^*(\mathbf{f}) \geq \frac{\phi(\mathbf{f})\, vol_\mathbf{f}(W) - \mathbf{e}^T_{S^* \to S^*}\boldsymbol{N}_\mathbf{f}\mathbf{f} - \mathbf{e}^T_{\bar{S^*} \to \bar{S^*}}\boldsymbol{N}_\mathbf{f}\mathbf{f}}{minvol_\mathbf{f}(S^*)},$$

Proof. The result is largely algebraic:

$$\begin{aligned}
\phi^*(\mathbf{f}) &= \frac{|\text{cut}_\mathbf{f}(S^*)|}{\text{minvol}_\mathbf{f}(S^*)} \\
&= \frac{(\mathbf{e}^T_{S^* \to \bar{S^*}} + \mathbf{e}^T_{\bar{S^*} \to S^*})\boldsymbol{N}_\mathbf{f}\mathbf{f}}{\text{minvol}_\mathbf{f}(S)} \\
&= \frac{\mathbf{e}^T\boldsymbol{N}_\mathbf{f}\mathbf{f} - \mathbf{e}^T_{S^* \to S^*}\boldsymbol{N}_\mathbf{f}\mathbf{f} - \mathbf{e}^T_{\bar{S^*} \to \bar{S^*}}\boldsymbol{N}_\mathbf{f}\mathbf{f}}{\text{minvol}_\mathbf{f}(S^*)} \\
&= \frac{\sum_{x \sim y}(\mathbf{f}(x) - \mathbf{f}(y)) - \mathbf{e}^T_{S^* \to S^*}\boldsymbol{N}_\mathbf{f}\mathbf{f} - \mathbf{e}^T_{\bar{S^*} \to \bar{S^*}}\boldsymbol{N}_\mathbf{f}\mathbf{f}}{\text{minvol}_\mathbf{f}(S)}.
\end{aligned}$$

Note that due to numbering of the vertices descending according to $\mathbf{f}$, $(\mathbf{f}(x) - \mathbf{f}(y))$ is always a non negative number. Moreover, using $\sum_{x \sim y}(\mathbf{f}(x) - \mathbf{f}(y)) = \sum_{i=1}^{n-1}(\mathbf{f}(x_i) - \mathbf{f}(x_{i+1}))|\text{cut}(S_i)|$ from [13], where S_i is the set of nodes labeled v_1 to v_i, we get:

$$\begin{aligned}
\phi^*(\mathbf{f}) &= \frac{\sum_{i=1}^{n-1}(\mathbf{f}(x_i) - \mathbf{f}(x_{i+1}))|\text{cut}(S_i)| - (\mathbf{e}^T_{S^* \to S^*} + \mathbf{e}^T_{\bar{S^*} \to \bar{S^*}})\boldsymbol{N}_\mathbf{f}\mathbf{f}}{\text{minvol}_\mathbf{f}(S^*)} \\
&\geq \frac{\phi(\mathbf{f}) \sum_{i=1}^{n-1}(\mathbf{f}(x_i) - \mathbf{f}(x_{i+1}))\ \text{minvol}(S_i) - \mathbf{e}^T_{S^* \to S^*}\boldsymbol{N}_\mathbf{f}\mathbf{f}}{\text{minvol}_\mathbf{f}(S^*)} \\
&\quad - \frac{\mathbf{e}^T_{\bar{S^*} \to \bar{S^*}}\boldsymbol{N}_\mathbf{f}\mathbf{f}}{\text{minvol}_\mathbf{f}(S^*)}.
\end{aligned}$$

As $\text{vol}(W) \leq \text{minvol}(S^*)$, we get:

$$\begin{aligned}
\phi^*(\mathbf{f}) &\geq \frac{\phi(\mathbf{f}) \sum_{x \in W} \mathbf{f}(x)\mathbf{d}(x) - \mathbf{e}^T_{S^* \to S^*}\boldsymbol{N}_\mathbf{f}\mathbf{f} - \mathbf{e}^T_{\bar{S^*} \to \bar{S^*}}\boldsymbol{N}_\mathbf{f}\mathbf{f}}{\text{minvol}_\mathbf{f}(S^*)} \\
&\geq \frac{\phi(\mathbf{f})\text{vol}_\mathbf{f}(W) - \mathbf{e}^T_{S^* \to S^*}\boldsymbol{N}_\mathbf{f}\mathbf{f} - \mathbf{e}^T_{\bar{S^*} \to \bar{S^*}}\boldsymbol{N}_\mathbf{f}\mathbf{f}}{\text{minvol}_\mathbf{f}(S^*)}.
\end{aligned}$$

□

Lemma 5.2. *We also have*

$$\phi^*(\mathbf{f}) \leq \phi^* \frac{minvol(S^*)}{minvol_\mathbf{f}(S^*)}.$$

Proof. By definition,

$$\phi^*(\mathbf{f}) = \frac{|\text{cut}_\mathbf{f}(S^*)|}{\text{minvol}_\mathbf{f}(S^*)}.$$

Because $\mathbf{f} \in [0,1]$, then the maximum value for $\mathbf{f}(i) - \mathbf{f}(j)$ is 1 for any edge (i,j) and therefore we get:

$$\phi^*(\mathbf{f}) \le \frac{|\text{cut}(S^*)|}{\text{minvol}_{\mathbf{f}}(S^*)} = \phi^* \frac{\text{minvol}(S^*)}{\text{minvol}_{\mathbf{f}}(S^*)}.$$

□

Theorem 5.3. *If we run a sweep-cut procedure on* $\mathbf{f}$ *where* $0 \le f_i \le 1$ *and* $vol(W) \le minvol(S^*)$*, then:*

$$\phi(\mathbf{f}) \le \phi^* \frac{minvol(S^*)}{vol_{\mathbf{f}} W} + \frac{\mathbf{e}^T_{S^* \to S^*} \boldsymbol{N}_{\mathbf{f}} \mathbf{f} + \mathbf{e}^T_{\bar{S^*} \to \bar{S^*}} \boldsymbol{N}_{\mathbf{f}} \mathbf{f}}{vol_{\mathbf{f}}(W)}.$$

Proof. By using the two previous lemmas and re-arranging the terms, we can prove the theorem. □

Note that the previous generalized sweepcut inequality depends on $\mathbf{f}$, W and S^*. If $\mathbf{f}$ is far from highlighting the set S^* or $\text{vol}(W)$ is small, then this bound will reduce to the trivial bound $\phi(\mathbf{f}) \le 1$. In the next section, we will use this generalized sweepcut inequality in addition to nonlinear diffusion properties to show that if the power function is used, then using small powers yields smaller conductance sets.

5.3 Implications for Nonlinear Diffusion

Lemma 5.4. *The sum property of nonlinear diffusion with growth or decay is:*

$$\mathbf{e}^T \mathbf{u}_t = \mathbf{e}^T \mathbf{u}_0 = 1.$$

Proof. By multiply both sides of the differential equation for nonlinear diffusion with growth or decay by $\mathbf{e}^T$ and using $\mathbf{e}^T \boldsymbol{L} = 0$, we get $\frac{\partial \mathbf{u}}{\partial t} = 0$. Therefore, the sum of the entries of $\mathbf{u}$ is always one. □

We will set $\mathbf{f} = g(\mathbf{u})$ and start our proofs using the general sweepcut inequality proposed in the previous section.

Lemma 5.5. *If* g *satisfies* $g(\mathbf{u}(x)) - g(\mathbf{u}(y)) \le \mathbf{u}(x) - \mathbf{u}(y)$ $\forall x, y \in W$*, then:*

$$\mathbf{e}^T_{S^* \to S^*} \boldsymbol{N}_{\mathbf{f}} \mathbf{f} + \mathbf{e}^T_{\bar{S^*} \to \bar{S^*}} \boldsymbol{N}_{\mathbf{f}} \mathbf{f} \le |W|.$$

Proof. By noting that $\mathbf{f}$ is nonzero only at the set W, we get:

$$\mathbf{e}^T_{S^* \to S^*} \boldsymbol{N}_{\mathbf{f}} \mathbf{f} + \mathbf{e}^T_{\bar{S^*} \to \bar{S^*}} \boldsymbol{N}_{\mathbf{f}} \mathbf{f} \le \mathbf{e}^T_{W \to W} \boldsymbol{N}_{\mathbf{f}} \mathbf{f}.$$

If g satisfies $g(\mathbf{u}(x)) - g(\mathbf{u}(y)) \le \mathbf{u}(x) - \mathbf{u}(y)$ $\forall x, y \in W$, then:

$$\begin{aligned}\mathbf{e}^T_{S^* \to S^*} \boldsymbol{N}_{\mathbf{f}} \mathbf{f} + \mathbf{e}^T_{\bar{S^*} \to \bar{S^*}} \boldsymbol{N}_{\mathbf{f}} \mathbf{f} &\le \mathbf{e}^T_{W \to W} \boldsymbol{N}_{\mathbf{f}} \mathbf{u} \\ &\le \sum_{x \in W} \mathbf{u}(x) |N(x) \cap W|.\end{aligned}$$

Here $N(x)$ is the set of nodes that are neighbors to x in the graph.

$$\mathbf{e}^T_{S^* \to S^*} \boldsymbol{N}_{\mathbf{f}} \mathbf{f} + \mathbf{e}^T_{\bar{S^*} \to \bar{S^*}} \boldsymbol{N}_{\mathbf{f}} \mathbf{f} \le \sum_{x \in W} \mathbf{u}(x) |W| = |W|.$$

□

Lemma 5.6.

$$vol_{\mathbf{f}}(W) \ge min_{i \in W} \mathbf{f}_i \, vol(W).$$

Proof.

$\text{vol}_{\mathbf{f}}(W) = \mathbf{e}^T_W \boldsymbol{D} \mathbf{f} \ge \min_{i \in W} \mathbf{f}_i \mathbf{e}^T_W \boldsymbol{D} \mathbf{e}_W = \min_{i \in W} \mathbf{f}_i \text{vol}(W).$

□

Theorem 5.7. *Sweepcut Inequality for Nonlinear Diffusion with growth or decay:*

If g *satisfies* $g(\mathbf{u}(x)) - g(\mathbf{u}(y)) \le \mathbf{u}(x) - \mathbf{u}(y)$ $\forall x, y \in W$ *and* $vol(W) \le minvol(S^*) \le v$*, then:*

$$\phi(\mathbf{f}) \le \phi^* \frac{v}{min_{i \in W} \mathbf{f}_i \, vol(W)} + \frac{|W|}{min_{i \in W} \mathbf{f}_i \, vol(W)}.$$

Proof. We can prove the theorem using the two previous lemmas. □

Theorem 5.8. *If* $minvol(S^*) \le v$*,* $g(\mathbf{u}) = \mathbf{u}^p$*,* p *tends to zero,* $\mathbf{d}_{avg}(W) \ge \frac{\alpha}{\phi^*}$*, and* $vol(W) \le minvol(S^*)$*, then the nonlinear diffusion sweepcut inequality becomes:*

$$\phi(\mathbf{f}) \le (\frac{v}{vol(W)} + \frac{1}{\alpha})\phi^*.$$

Proof. By noting that, as p tends to zero, $\mathbf{u}$ tends to an indicator and therefore, the condition $g(\mathbf{u}(x)) - g(\mathbf{u}(y)) \le \mathbf{u}(x) - \mathbf{u}(y) \forall x, y \in W$ is always satisfied. In this case, $\mathbf{f}$ will tend to be $\mathbf{e}_W$ and the following properties applies for $\mathbf{e}_W$. $\min_{i \in W} \mathbf{f}_i \text{vol}(W)$ will tend to $\text{vol}(W)$ and we can observe that:

$$\frac{|W|}{\min_{i \in W} \mathbf{f}_i \text{vol}(W)} = \frac{|W|}{\text{vol}(W)} = \frac{|W|}{|W| \mathbf{d}_{\text{avg}}(W)} = \frac{1}{\mathbf{d}_{\text{avg}}(W)}.$$

As $\mathbf{d}_{\text{avg}}(W) \ge \frac{\alpha}{\phi^*}$ and re-arranging the terms, we can prove the theorem. Note that we can control $\text{vol}(W)$ using k, the number of steps, as increasing k increases $\text{vol}(W)$. □

At its heart, this theorem states that if the result of the diffusion is close to an indicator vector, then we will have some relationship between the set of smallest conductance and the result. This suggests using the power function with small power as it will tend to encourage "indicator-like" behavior for any number of time-steps. If the graph has an average degree of $\Omega(\frac{1}{\phi^*})$, then the nonlinear diffusion conductance will be close to the smallest conductance in the graph by a multiplicative factor and the larger the average degree of the graph, the smaller the multiplicative factor. Additionally, we can use this sweepcut inequality to tell us how close the nonlinear diffusion conductance is to the smallest conductance in the graph by simply computing the constants. On the other hand, this theorem only provides coarse guidance and essentially requires a large degree of knowledge of the optimal set to yield non-trivial results.

6 EXPERIMENTS

The goal of our experiments is to show the advantage of our nonlinear diffusion process in a variety of settings. We first show that it gives better F1-values in LFR synthetic graphs compared with linear diffusions, where we get the best results with small p as hinted at by our theory results. Second, in community detection on SNAP graphs, the non-linear diffusion gives consistently higher $F1$ scores on all graphs except one. Third, on graph-based semi-supervised learning, we show that nonlinear diffusion improves substantially on recently developed methods based on neural networks in two of three tests. Finally, we conclude our experiments by trying to understand why nonlinear diffusion gives better results. Thus, we conduct a case study in the Fashion MNIST data and argue that the reason nonlinear diffusions perform better is because they produce vectors with higher dispersion than linear diffusions. Our experimental code is available at: https://github.com/RaniaSalama/Nonlinear_Diffusion

6.1 Synthetic Data

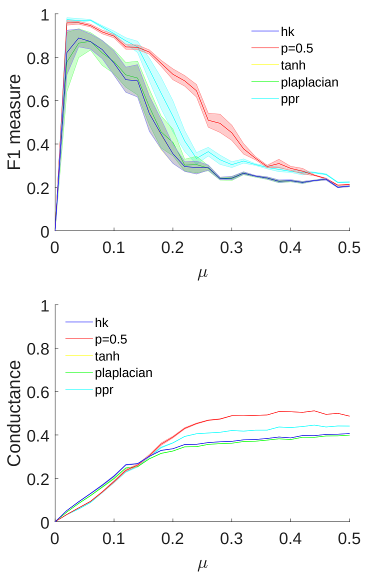


Figure 2: The F1 measure and the conductance of various techniques on LFR synthetic datasets. The bands show the variation over 100 trials.

We use the LFR model [28] to generate the synthetic datasets as it is a widely used model in evaluating community detection algorithms. The parameters used for the model are $n = 1000$, average degree is 10, maximum degree is 50, minimum community size is 20 and maximum community size is 100. LFR node degrees and community sizes are distributed according to the power law distribution, we set the exponent for the degree sequence to 2 and the exponent for the community size distribution to 1. We vary μ the mixing parameter from 0.02 to 0.5 with step 0.02 and run the heat kernel diffusion (hk), personalized PageRank (ppr), nonlinear diffusion using power function with $p = 0.5$, nonlinear diffusion using tanh function and nonlinear diffusion using p-Laplacian. In all the experiments, we set $k = 100$ and $h = 0.001$. For each graph, we start from a random seed node and we repeat the experiment 100 times. Figure 2 shows the mean and the variance for the F1 measure and the conductance while varying μ. As shown in the figure, nonlinear diffusion using power function with $p = 0.5$ has a better F1 measure than the heat kernel diffusion and has lower conductance than it for $\mu = 0.02$ to 0.14. The conductance for the nonlinear diffusion gets slightly higher after $\mu = 0.14$ but the increase in F1 measure is much higher. For example, for $\mu = 0.24$, the increase in conductance of $p = 0.5$ over the heat kernel diffusion is $\approx 9\%$, while the increase in F1 measure is $\approx 30\%$.

6.2 Local Community Detection

Local community detection is the task of finding the community when given a member of that community. In this task, we start the diffusion from the given node. Table 1 shows the community detection results for heat kernel diffusion (hk), personalized PageRank (ppr), nonlinear diffusion using the power function as the nonlinear function (power) with $p = 0.5$, nonlinear diffusion using the tanh function (tanh) and nonlinear diffusion using the p-Laplacian operator (p-Lap) with $p = 1.9$. In all nonlinear diffusions, we set $k = 10, h = 0.001$. We have followed the same evaluation settings as mentioned in [27]. In these experiments, we identify 100 communities from the ground truth such that each community has a size greater than 10, then for all algorithms, we start from each node in the community and finally report the result from the node that yields the best F1 measure. For nonlinear diffusion we truncate the diffusion values that are less than ε to be zero and the diffusion values that are greater than one to be one, where ε is designed to give 100 nodes in the final community, where we grow larger if necessary.

Table 1 shows datasets from SNAP repository [30], their characteristics, the mean F1 measure and the mean set size for the heat kernel diffusion, the personalized PageRank and the nonlinear diffusions. These results show that nonlinear diffusion can return a larger set than the heat kernel diffusion with higher or comparable F1 score. For example, in YouTube dataset, the detected communities for nonlinear diffusion have a higher F1 measure by around 35%.

6.3 Graph-based Semi-supervised Learning

Semi-supervised learning is the task of learning from a small set of labeled examples and a very large set of unlabeled examples. In these experiments, we use three citation networks, which are Cora, Citeseer and Pubmed. Table 2 shows the characteristic of these datasets. We run two coupled nonlinear diffusions as in Section 3.2, one using the adjacency matrix and another one using an r nearest neighbor graph constructed from the feature matrix using radial basis function with width equal 1. Following the exact methodology of [26], we choose the hyperparameters based on a train-validate-test split where we pick the parameters based on the ones that have the best classification accuracy on the validation set and then show the evaluation on the test set. The parameters are the number of nearest neighbors r, the number of steps k, the value of the power function p_1 for the adjacency diffusion and p_2 for the nearest neighbor diffusion, the value of σ, and value or rank rounding. For reproducibility, the parameters we find are given in a footnote.[6]

[6]For all the experiments, we set $h = 0.001$. Let r be the number of nearest neighbors used in the graph construction. For the power function, in Cora, we set the $r = 100$, $k = 400$, $\sigma = 0.55$, $p_1 = 0.6$, $p_2 = 0.45$ and used value-based rounding; in Citeseer we set $r = 200$, $k = 400$, $\sigma = 0.51$, $p_1 = 0.55$, $p_2 = 0.55$ and used rank-based rounding; and in Pubmed we set $r = 2000$, $k = 300$, $\sigma = 0.65$, $p_1 = 0.7$, $p_2 = 0.6$ and used rank-based rounding. For nonlinear diffusion using tanh function, in Cora we set $r = 100$, $k = 450$, $\sigma = 1.9$ and used value-based rounding; in Citeseer we set $r = 200$, $k = 450$, $\sigma = 1.9$ and used rank-based rounding; and in Pubmed wet set $r = 2000$, $k = 400$, $\sigma = 1.8$ and used rank-based rounding. Finally, for p-Laplacian, for Cora, we set $r = 100$, $p_1 = 1.9$, $p_2 = 1.7$, $\sigma = 0.05$, $k = 400$; for Citeseer,

Table 1: Community detection results.

	|V|	|E|	F1 measure					Set size				
			hk	ppr	power	tanh	p-Lap	hk	ppr	power	tanh	p-Lap
DBLP	317,080	1,049,866	0.364	0.273	**0.462**	0.401	0.403	156	4,529	599.59	113.5	136.21
Amazon	334,863	925,872	**0.608**	0.415	0.559	0.505	0.476	145	5,073	479.76	34.70	74.20
YouTube	1,134,890	2,987,624	0.128	0.078	**0.470**	0.388	0.422	137	5,833	89.46	680.60	215.82
LiveJournal	3,997,962	34,681,189	0.138	0.104	0.251	0.258	**0.269**	156	299	104.29	773.35	717.49

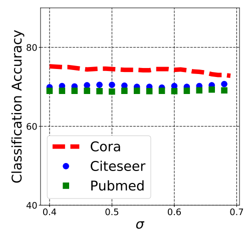
Table 2: Characteristics of datasets used in graph-based semi-supervised learning experiments.

	|V|	|E|	|Classes|	d	Label rate
Citeseer	3,327	4,732	6	3,703	0.036
Cora	2,708	5,429	7	1,433	0.052
Pubmed	19,717	44,338	3	500	0.003

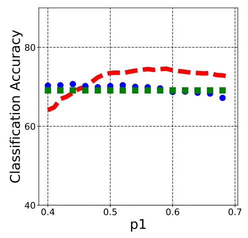
Table 3: Classification accuracy for graph-based semi-supervised learning experiments.

	Cora	Citeseer	Pubmed
ManiReg [3]	59.5	60.1	70.7
SemiEmb [53]	59.0	59.6	71.1
LP [58]	68.0	45.3	63.0
DeepWalk [39]	67.2	43.2	65.3
ICA [33]	75.1	69.1	73.9
Planetoid [56]	75.7	64.7	77.2
GCN [26]	81.5	70.3	**79.0**
GCN + self learning [31]	80.2	67.8	76.9
hk	61.4	53.8	63.3
power	74.2	70.6	70.0
power + self-learning	71.3	68.9	70.3
tanh	77.9	**73.4**	71.0
tanh + self-learning	**82.9**	72.2	73.0
p-Lap	79.8	68.7	73.4
p-Lap + self-learning	79.2	67.8	73.4

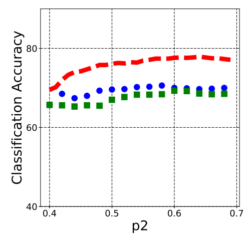
Table 3 shows the classification accuracy of nonlinear diffusion compared to other graph-based semi-supervised learning related work. As shown in the table nonlinear diffusion using tanh function has the best classification accuracy in Cora and Citeseer. However, GCN has the best classification accuracy in Pubmed. Note that,



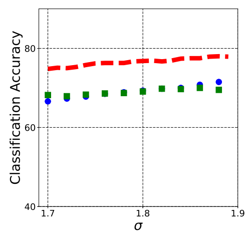
(a) Varying σ



(b) Varying p_1



(c) Varying p_2



(d) Varying σ in a tanh diffusion

Figure 3: As we vary parameters, the accuracy remains high for many choices.

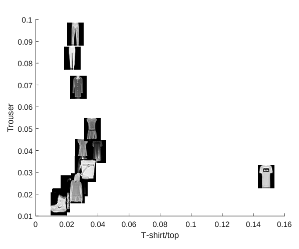
nonlinear diffusion uses less parameters as we do no optimization over the weights of the layers and therefore it is the simplest model with good accuracy. Additionally, the dynamics of the nonlinear diffusion is described by a differential equation and therefore it is easier to reason about and to develop bounds for its performance.

To further analyze the stability of nonlinear diffusion when the parameters change, Figure 3 shows changing σ, p_1 and p_2 for nonlinear diffusion using power function from 0.4 to 0.7 with step of 0.02 and changing σ for nonlinear diffusion using tanh function from 1.7 to 1.9 with step 0.02. As shown in the figure, nonlinear diffusion is robust to small changes in the parameters.

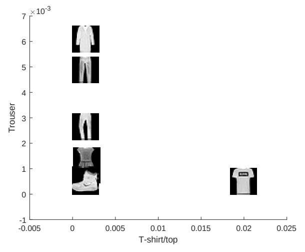
6.4 Fashion MNIST Case Study

Our final experiment is a case study of the Fashion MNIST dataset to understand why nonlinear diffusion outperforms linear diffusion. The Fashion MNIST dataset is a collection of Zalando's clothing images [54] with 70,000, 28 by 28 grayscale images. Each example is assigned a label from 10 classes, which are: T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag and Ankle Boot.

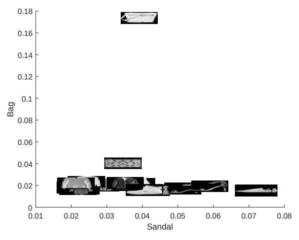
we set $r = 300$, $p_1 = 1.8$, $p_2 = 1.7$, $\sigma = 0.7$, $k = 1000$; and for Pubmed, we set $r = 100$, $p_1 = 1.9$, $p_2 = 1.7$, $\sigma = 0.05$, $k = 400$.



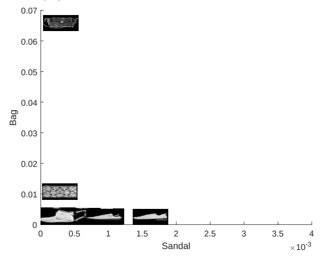
(a) Nonlinear Diffusion



(b) Heat Kernel Diffusion



(c) Nonlinear Diffusion



(d) Heat Kernel Diffusion

Figure 4: Comparison between the heat kernel diffusion and the nonlinear diffusion on Fashion-MNIST dataset.

Table 4: Heat kernel diffusion and nonlinear diffusion error rates on Fashion-MNIST dataset (k = 1000, h = 0.001, labels per class = 10, and r = 100).

	hk	power
T-shirt/top	0.466 ± 0.211	**0.362 ± 0.228**
Trouser	0.180 ± 0.084	**0.061 ± 0.017**
Pullover	0.609 ± 0.140	**0.582 ± 0.208**
Dress	0.547 ± 0.171	**0.481 ± 0.275**
Coat	0.516 ± 0.199	**0.502 ± 0.220**
Sandal	0.546 ± 0.165	**0.499 ± 0.256**
Shirt	**0.795 ± 0.109**	0.820 ± 0.098
Sneaker	0.217 ± 0.116	**0.196 ± 0.137**
Bag	0.528 ± 0.118	**0.403 ± 0.130**
Ankle boot	0.325 ± 0.138	**0.188 ± 0.190**
Total	0.473 ± 0.030	**0.409 ± 0.040**

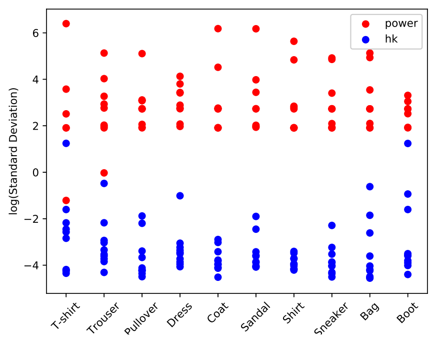


Figure 5: Dispersion comparison between the heat kernel diffusion values and nonlinear diffusion.

We construct a r-nearest neighbor graph from the data. Then, we randomly select 10 examples per class to start the semi-supervised learning task. Finally, we repeat the experiments 10 times and report the error rate over the 70,000 examples. This means that we only use around 0.14% of examples for the learning task.

Table 4 shows the error rates for the heat kernel diffusion and the nonlinear diffusion using power function with $p = 0.5$ for each different classes. These results show that nonlinear diffusion has a lower error rate than the heat kernel diffusion in all class except the Shirt class. Additionally, overall the error rate for the nonlinear diffusion was lower by around 7%.

To further study the difference between the two techniques, Figure 4 (a, b) compares values of the diffusion as in the introduction (Figure 1). Specifically, these figures are run on a random sample of 1000 images where we start from a single T-shirt/top example in the x-axis and the diffusion values when we start from a single coat example in the y-axis. Then we show 2 randomly chosen pictures from each class (so 20 pictures total). As shown in the figures, the heat kernel diffusion is extremely spiky for the different examples, while nonlinear diffusion is able to distinguish between different examples to provide more information to downstream operators. Figure 4 (c, d) shows the same comparison for Sandals and Bags.

We did one final evaluation to understand this advantage. Our conjecture is that the nonlinear diffusion produces values with far more dispersion over the range than the heat kernel. In Figure 5, we draw the log of the standard deviation of the diffusion values when we start the diffusion from each class for each trial. As shown in the figure, the nonlinear diffusion values show much higher standard deviations than the heat kernel diffusion, which supports this point.

7 DISCUSSION AND FUTURE WORK

In this paper, we have shown that nonlinear diffusion is competitive with embedding techniques and significantly outperform classical diffusions in community detection and semi-supervised learning. As a simple diffusion model, there are ample opportunities to accelerate the computation using a wide variety of techniques and we have not made that a focus of our work. For instance, there are Monte Carlo [14] and localized algorithms [2, 27] for linear diffusions that yield sublinear complexity and millisecond-level runtimes on billion edge graphs. In comparison, the complexity of our current implementation is linear (the work is k matrix-vector products). Note that, in our experience, the time required to evaluate these diffusions is dominated by the initial data load, ingestion, and r-nearest neighbor computation so we do not view this as problematic, but for large numbers of classes, faster runtime may be useful. We also note that our approach can easily be combined with recent work on higher-order analysis of graphs based on motifs [5]. Specific things we will explore in the future include how to choose the nonlinear function g in an automated fashion by studying the properties of target sets in the graph. Additionally, we hope to provide improved algorithms to evaluate the nonlinear ODEs themselves.

Acknowledgments. Both authors would like to acknowledge our colleagues who provided initial feedback on this paper. This work is supported by NSF awards CCF-1149756, IIS-1422918, IIS-1546488; NSF Center for Science of Information STC, CCF-0939370, DARPA SIMPLEX, and the Sloan Foundation.

REFERENCES

[1] S. Amghibech. 2003. Eigenvalues of the Discrete p-Laplacian for Graphs. *Ars Comb.* 67 (2003).

[2] Reid Andersen, Fan Chung, and Kevin Lang. 2006. Local Graph Partitioning using PageRank Vectors. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*. IEEE, IEEE, Washington, D.C., USA, 475–486. http://www.math.ucsd.edu/~fan/wp/localpartition.pdf

[3] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. 2006. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research* 7, Nov (2006), 2399–2434.

[4] Philippe Bénilan and Michael G Crandall. 1981. The continuous dependence on φ of solutions of u t- $\Delta\varphi$ (u)= 0. *Indiana University Mathematics Journal* 30, 2 (1981), 161–177.

[5] Austin Benson, David F. Gleich, and Jure Leskovec. 2016. Higher-order organization of complex networks. Submitted. *Science* 353, 6295 (2016), 163–166. https://doi.org/10.1126/science.aad9029

[6] James G Berryman and Charles J Holland. 1978. Nonlinear diffusion problem arising in plasma physics. *Physical Review Letters* 40, 26 (1978), 1720.

[7] M. Bonforte, J. Dolbeault, G. Grillo, and J. L. Vazquez. 2010. Sharp rates of decay of solutions to the nonlinear fast diffusion equation via functional inequalities. *Proceedings of the National Academy of Sciences* 107, 38 (sep 2010), 16459–16464. https://doi.org/10.1073/pnas.1003972107

[8] Nick Bridle and Xiaojin Zhu. 2013. p-voltages: Laplacian regularization for semi-supervised learning on high-dimensional data. In *Eleventh Workshop on Mining and Learning with Graphs (MLG2013)*. ACM, New York, NY, USA.

[9] Thomas Bühler and Matthias Hein. 2009. Spectral clustering based on the graph p-Laplacian. In *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, ACM, New York, NY, USA, 81–88.

[10] O Chapelle and B Sch. 2006. olkopf, and A. Zien, Semi-Supervised Learning, vol. 2.

[11] Fan Chung. 2005. Laplacians and the Cheeger inequality for directed graphs. *Annals of Combinatorics* 9, 1 (2005), 1–19.

[12] Fan Chung. 2007. Four proofs for the Cheeger inequality and graph partition algorithms. In *Proceedings of ICCM*, Vol. 2. ScienTech, Mason, OH, USA, 378.

[13] Fan Chung. 2007. The heat kernel as the pagerank of a graph. *Proceedings of the National Academy of Sciences* 104, 50 (2007), 19735–19740.

[14] Fan Chung and Olivia Simpson. 2013. Solving Linear Systems with Boundary Conditions Using Heat Kernel Pagerank. In *Algorithms and Models for the Web Graph*. Springer, 203–219.

[15] Fan R. L. Chung. 1992. *Spectral Graph Theory*. American Mathematical Society, Providence, RI, USA.

[16] George Cybenko. 1989. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems* 2, 4 (1989), 303–314.

[17] Rumi Ghosh, Shang-hua Teng, Kristina Lerman, and Xiaoran Yan. 2014. The interplay between dynamics and networks: centrality, communities, and cheeger inequality. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, ACM, New York, NY, USA, 1406–1415.

[18] David F. Gleich. 2015. PageRank beyond the Web. *SIAM Rev.* 57, 3 (August 2015), 321–363. https://doi.org/10.1137/140976649

[19] David F Gleich and Michael W Mahoney. 2015. Using local spectral methods to robustify graph-based learning algorithms. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, ACM, New York, NY, USA, 359–368.

[20] David F. Gleich and Michael W. Mahoney. 2016. Mining large graphs. In *Handbook of Big Data*, Peter Bühlmann, Petros Drineas, Michael Kane, and Mark van de Laan (Eds.). CRC Press, Boca Raton, Florida, USA, 191–220. https://doi.org/10.1201/b19567-17

[21] David F. Gleich and Ryan A. Rossi. 2014. A Dynamical System for PageRank with Time-Dependent Teleportation. *Internet Mathematics* 10, 1–2 (June 2014), 188–217. https://doi.org/10.1080/15427951.2013.814092

[22] Aditya Grover and Jure Leskovec. 2016. Node2Vec: Scalable Feature Learning for Networks. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. ACM, New York, NY, USA, 855–864. https://doi.org/10.1145/2939672.2939754

[23] FM Henderson and RA Wooding. 1964. Overland flow and groundwater flow from a steady rainfall of finite duration. *Journal of Geophysical Research* 69, 8 (1964), 1531–1540.

[24] Thorsten Joachims. 2003. Transductive Learning via Spectral Graph Partitioning. In *ICML*. ACM, New York, NY, USA, 290–297. http://www.aaai.org/Papers/ICML/2003/ICML03-040.pdf

[25] J. R. KING. 1988. Extremely High Concentration Dopant Diffusion in Silicon. *IMA Journal of Applied Mathematics* 40, 3 (1988), 163–181. https://doi.org/10.1093/imamat/40.3.163

[26] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* abs/1609.02907 (2016).

[27] Kyle Kloster and David F Gleich. 2014. Heat kernel based community detection. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, ACM, New York, NY, USA, 1386–1395.

[28] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. 2008. Benchmark graphs for testing community detection algorithms. *Physical review E* 78, 4 (2008), 046110.

[29] LS Leibenzon. 1930. The motion of a gas in a porous medium. Complete Works, Vol. 2, Acad. *Sciences URSS, Moscow,(Russian)* 63 (1930), 8–9.

[30] Jure Leskovec and Andrej Krevl. 2014. {SNAP Datasets}:{Stanford} Large Network Dataset Collection. http://snap.stanford.edu/data.

[31] Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper Insights Into Graph Convolutional Networks for Semi-Supervised Learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018.* AAAI Press, Palo Alto, CA, USA, 3538–3545.

[32] Yixuan Li, Kun He, David Bindel, and John E. Hopcroft. 2015. Uncovering the Small Community Structure in Large Networks: A Local Spectral Approach. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15)*. International World Wide Web Conferences Steering Committee, Geneva, Switzerland, 658–668. https://doi.org/10.1145/2736277.2741676

[33] Qing Lu and Lise Getoor. 2003. Link-based classification. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*. ACM, New York, NY, USA, 496–503.

[34] Ulrike Luxburg. 2007. A tutorial on spectral clustering. *Statistics and Computing* 17, 4 (December 2007), 395–416. https://doi.org/10.1007/s11222-007-9033-z

[35] Michael W. Mahoney, Lorenzo Orecchia, and Nisheeth K. Vishnoi. 2012. A Local Spectral Method for Graphs: With Applications to Improving Graph Partitions and Exploring Data Graphs Locally. *Journal of Machine Learning Research* 13 (August 2012), 2339–2365. http://www.jmlr.org/papers/volume13/mahoney12a/mahoney12a.pdf

[36] Boaz Nadler, Nathan Srebro, and Xueyuan Zhou. 2009. Semi-supervised learning with the graph laplacian: The limit of infinite unlabelled data. *Advances in neural information processing systems* 22 (2009), 1330–1338.

[37] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank Citation Ranking: Bringing Order to the Web.* Technical Report 1999-66. Stanford University. http://dbpubs.stanford.edu:8090/pub/1999-66

[38] Jia-Yu Pan, Hyung-Jeong Yang, Christos Faloutsos, and Pinar Duygulu. 2004. Automatic multimedia cross-modal correlation discovery. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, New York, NY, USA, 653–658. https://doi.org/10.1145/1014052.1014135

[39] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, ACM, New York, NY, USA, 701–710.

[40] A. P. Riascos and José L. Mateos. 2014. Fractional dynamics on networks: Emergence of anomalous diffusion and Lévy flights. *Physical Review E* 90, 3 (sep 2014). https://doi.org/10.1103/physreve.90.032809

[41] Sam T Roweis and Lawrence K Saul. 2000. Nonlinear dimensionality reduction by locally linear embedding. *science* 290, 5500 (2000), 2323–2326.

[42] F. Scarselli, M. Gori, Ah Chung Tsoi, M. Hagenbuchner, and G. Monfardini. 2009. The Graph Neural Network Model. *IEEE Transactions on Neural Networks* 20, 1 (jan 2009), 61–80. https://doi.org/10.1109/tnn.2008.2005605

[43] Satu Elisa Schaeffer. 2007. Graph clustering. *Computer Science Review* 1, 1 (2007), 27–64. https://doi.org/10.1016/j.cosrev.2007.05.001

[44] Lawrence F Shampine, Skip Thompson, JA Kierzenka, and GD Byrne. 2005. Non-negative solutions of ODEs. *Appl. Math. Comput.* 170, 1 (2005), 556–569.

[45] Dane Taylor, Sean A. Myers, Aaron Clauset, Mason A. Porter, and Peter J. Mucha. 2017. Eigenvector-Based Centrality Measures for Temporal Networks. *Multiscale Modeling & Simulation* 15, 1 (jan 2017), 537–574. https://doi.org/10.1137/16m1066142

[46] Joshua B Tenenbaum, Vin De Silva, and John C Langford. 2000. A global geometric framework for nonlinear dimensionality reduction. *science* 290, 5500 (2000), 2319–2323.

[47] JL Vázquez. 2005. Smoothing and decay estimates for nonlinear parabolic equations of porous medium type. *preprint* (2005).

[48] Juan Luis Vázquez. 2007. Perspectives in nonlinear diffusion: between analysis, physics and geometry. In *Proceedings of the International Congress of Mathematicians Madrid, August 22–30, 2006*. American Mathematical Society, Providence, RI, USA, 609–634.

[49] Juan Luis Vázquez. 2017. The Mathematical Theories of Diffusion: Nonlinear and Fractional Diffusion. In *Nonlocal and Nonlinear Diffusions and Interactions: New Methods and Directions*. Springer International Publishing, New York, NY, USA, 205–278. https://doi.org/10.1007/978-3-319-61494-6_5

[50] Luke N. Veldt, David F. Gleich, and Michael W. Mahoney. 2016. A Simple and Strongly-Local Flow-Based Method for Cut Improvement. In *International Conference on Machine Learning*. ACM, New York, NY, USA, 1938–1947. http://jmlr.org/proceedings/papers/v48/veldt16.html

[51] James Voss, Mikhail Belkin, and Luis Rademacher. 2016. The Hidden Convexity of Spectral Clustering. In *AAAI*. AAAI Press, Palo Alto, CA, USA, 2108–2114. https://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12307/11849

[52] Di Wang, Kimon Fountoulakis, Monika Henzinger, Michael W. Mahoney, and Satish Rao. 2017. Capacity Releasing Diffusion for Speed and Locality. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Doina Precup and Yee Whye Teh (Eds.), Vol. 70. PMLR, International Convention Centre, Sydney, Australia, 3598–3607. http://proceedings.mlr.press/v70/wang17b.html

[53] Jason Weston, Frédéric Ratle, Hossein Mobahi, and Ronan Collobert. 2012. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*. Springer, New York, NY, USA, 639–655.

[54] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *CoRR* abs/1708.07747 (2017).

[55] Xiaoran Yan, Shang-hua Teng, Kristina Lerman, and Rumi Ghosh. 2016. Capturing the interplay of dynamics and networks through parameterizations of Laplacian operators. *PeerJ Computer Science* 2 (2016), e57.

[56] Zhilin Yang, William Cohen, and Ruslan Salakhudinov. 2016. Revisiting Semi-Supervised Learning with Graph Embeddings. In *International Conference on Machine Learning*. ACM, New York, NY, USA, 40–48.

[57] Denny Zhou, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf. 2004. Learning with local and global consistency. In *Advances in neural information processing systems*. MIT Press, Cambridge, MA, USA, 321–328.

[58] Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. 2003. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*. ACM, New York, NY, USA, 912–919.