New Insights into Laplacian Similarity Search

Xiao-Ming Wu[1] Zhenguo Li[2] Shih-Fu Chang[1]
[1]Department of Electrical Engineering, Columbia University
[2]Huawei Noah's Ark Lab, Hong Kong
{xmwu,sfchang}@ee.columbia.edu li.zhenguo@huawei.com

Abstract

Graph-based computer vision applications rely critically on similarity metrics which compute the pairwise similarity between any pair of vertices on graphs. This paper investigates the fundamental design of commonly used similarity metrics, and provides new insights to guide their use in practice. In particular, we introduce a family of similarity metrics in the form of $(L + \alpha\Lambda)^{-1}$, where L is the graph Laplacian, Λ is a positive diagonal matrix acting as a regularizer, and α is a positive balancing factor. Such metrics respect graph topology when α is small, and reproduce well-known metrics such as hitting times and the pseudo-inverse of graph Laplacian with different regularizer Λ.

This paper is the first to analyze the important impact of selecting Λ in retrieving the local cluster from a seed. We find that different Λ can lead to surprisingly complementary behaviors: $\Lambda = D$ (degree matrix) can reliably extract the cluster of a query if it is sparser than surrounding clusters, while $\Lambda = I$ (identity matrix) is preferred if it is denser than surrounding clusters. Since in practice there is no reliable way to determine the local density in order to select the right model, we propose a new design of Λ that automatically adapts to the local density. Experiments on image retrieval verify our theoretical arguments and confirm the benefit of the proposed metric. We expect the insights of our theory to provide guidelines for more applications in computer vision and other domains.

1. Introduction

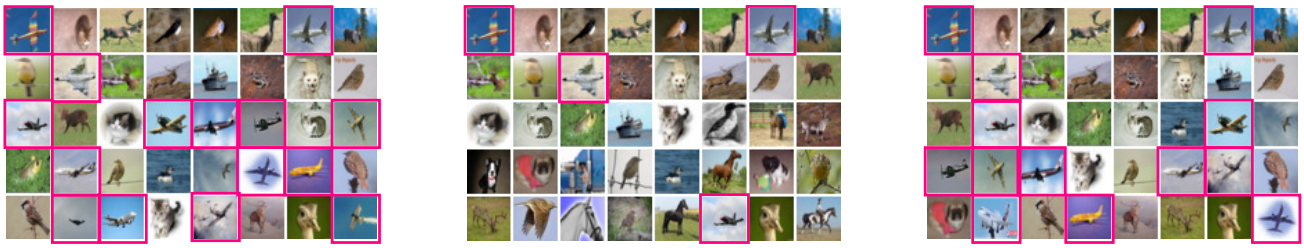
Quite a few computer vision applications have been successfully modeled on graphs, including image segmentation [21, 3, 7, 28, 15], image retrieval [29, 8, 5, 16, 10], manifold learning [24, 14, 27, 2, 22], and face recognition [9]. A central element in various graph-based applications is the notion of similarity between vertices. Well-known similarity metrics include personalized PageRank [20], hitting and commute times [17], and the pseudo-inverse of graph Laplacian [6]. Despite their popularity, the understanding of their behaviors is far from complete, and their use in practice is mostly guided by empirical trials and error analysis. This paper bridges this gap by investigating the fundamental design of commonly used similarity metrics, revealing their hidden assumptions, characterizing their behaviors, and solving the model selection problem.

We propose to study a family of graph similarity metrics in the form of $M = (L + \alpha\Lambda)^{-1}$, where L is the graph Laplacian matrix, Λ is a positive diagonal matrix acting as a regularizer, and α is a positive balancing factor controlling the degree of regularization. Our motivations are as follows. First, unlike metrics such as commute times which do not capture the global graph structure in large random neighborhood graphs [23], M is shown to respect the graph structure when α is small [25]. Second, as shown in this paper, given a vertex as query, ranking by M converges to a meaningful limit when $\alpha \to 0$ (see Sec. 2). In contrast, other metrics with balancing parameters such as personalized PageRank [20] and manifold ranking [29] converge to uninformative results that are solely determined by vertex degrees, making it sensitive to select the appropriate balancing parameter. Third, we will show that M unifies several well-known metrics with different Λ. For example, with $\Lambda = D$ (degree matrix), M corresponds to the hitting times and the degree normalized personalized PageRank [1]; and with $\Lambda = I$ (identity matrix), M corresponds to the the pseudo-inverse of graph Laplacian. This unified view opens the door to compare and analyze these otherwise irrelevant metrics under a single framework.

One of the main contributions of this paper is to provide a unified analysis to reveal the important impact of selecting the regularizer Λ in retrieving the local cluster from a seed. We find that different Λ can lead to surprisingly complementary behaviors. $\Lambda = D$ works amazingly well in extracting the cluster sparser than its surrounding clusters but is less preferred than $\Lambda = I$ the other way around. On the contrary, $\Lambda = I$ is particularly desirable if the cluster is denser than its surrounding clusters but is much worse than $\Lambda = D$ otherwise. These findings allow us to select the better metric for retrieval if the density of the cluster of interest

(a) $\Lambda = I$, AP = 0.14 (b) $\Lambda = D$, AP = 0.67 (c) $\Lambda = H$, AP = 0.67

Figure 1. The top 40 retrieved images on the extended YaleB dataset (query is on top left). False images are bounded by blue boxes.



(a) $\Lambda = I$, AP = 0.27 (b) $\Lambda = D$, AP = 0.17 (c) $\Lambda = H$, AP = 0.27

Figure 2. The top 40 retrieved images on the CIFAR-10 dataset (query is on top left). Positive images are bounded by magenta boxes.

is known ahead.

However, in practice there is no reliable way to automatically detect the local density in order to select the right model. Another contribution of this paper is to tackle this important model selection problem. We propose a new metric ($\Lambda = H$, see Sec. 3) which integrates the strengths of the above two complementary metrics. $\Lambda = H$ is desirable in that it behaves like $\Lambda = D$ when the query lies in a sparse cluster, while behaves like $\Lambda = I$ when the query comes from a dense cluster. The benefit of $\Lambda = H$ is justified by our theoretical analysis and experiments on synthetic and real datasets.

The key message of this paper can be summarized visually in the image retrieval results in Figs. 1 and 2. In Fig. 1, the query image comes from the sparsest cluster of total 38 clusters in the extended YaleB dataset [13], while in Fig. 2, the query image is from the densest cluster of total 10 clusters in the CIFAR-10 dataset [11]. As expected, $\Lambda = D$ and $\Lambda = I$ show distinctive yet complementary behaviors, while $\Lambda = H$ automatically biases to the better of the two. More experiments on image retrieval, including the dataset description and parameter setup, are reported in Sec. 4.

While we only report experiments on image retrieval, our theories and results apply to any visual application that relies on similarity measure and we expect them to guide more visual applications in the future.

2. The Laplacian-Based Similarity Metrics

Throughout the paper, we consider weighted graphs which are connected and undirected without self-loops. Let $\mathcal{G} = (\mathcal{V}, W)$ be a graph with a set $\mathcal{V}$ of n vertices and a symmetric non-negative affinity matrix $W = [w_{ij}] \in \mathbb{R}^{n\times n}$ ($w_{ii} = 0$). The degree of vertex i is $d_i = \sum_j w_{ij}$. $D = \mathrm{diag}(d_1, d_2, \cdots, d_n)$ is called the degree matrix of the graph. The graph Laplacian [4] is $L = D - W$, and the symmetric normalized graph Laplacian is $L_{sym} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$. Denote by $\Lambda = \mathrm{diag}(\lambda_1, \cdots, \lambda_n)$ an arbitrary positive diagonal matrix.

We consider similarity metrics in the following form:

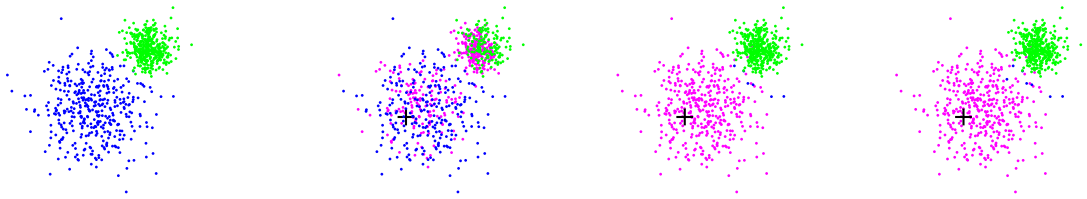
$$M = [m_{ij}] \in \mathbb{R}^{n\times n} = (L + \alpha\Lambda)^{-1}, \qquad (1)$$

where Λ is a positive diagonal matrix and α is a positive balancing factor. Note that while L is degenerate, $L + \alpha\Lambda$ is invertible, where Λ acts as a regularizer and α controls the degree of regularization. As M is derived based on the graph Laplacian L, we call M the Laplacian-based similarity metric. In this section, we present the desirable properties of M as graph similarity metrics, and show how it unifies existing metrics. We also provide examples and intuitions to demonstrate the role of the regularizer Λ in capturing the graph structures. Due to space limit, all proofs in this paper are included in the supplement.

2.1. Basic Properties

As a similarity matrix M, m_{ij} characterizes certain closeness between vertices i and j. The larger m_{ij} is, the closer i and j are. M is positive and symmetric, i.e., $\forall i, j$, $m_{ij} > 0$, and $m_{ij} = m_{ji}$. Regardless of Λ, m_{ii} is always the unique largest element in the i-th column and row of M (see supplement). As shown in Figs. 1 and 2, the query image is always ranked first regardless of Λ.

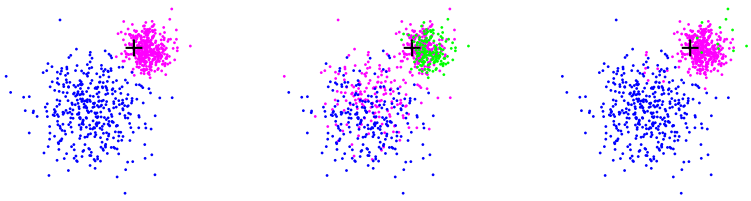
The scalar α plays a role of balancing the graph Laplacian L and the regularizer Λ. When α is small, the columns



(a) Two Gaussians (b) $\Lambda = I$ (c) $\Lambda = D$ (d) $\Lambda = H$

	Sparse	Dense	All
I	.5403	**.9992**	.7698
D	**.9888**	.7888	.8888
H	**.9720**	**.9516**	**.9618**

(e) MAP



(f) $\Lambda = I$ (g) $\Lambda = D$ (h) $\Lambda = H$

Figure 3. Two 20-dimensional Gaussians with variances 1 and 0.16, and 400 points in each. The black cross denotes a query. The top 400 ranked points are highlighted in magenta. (d&h) Ranking by our proposed H (Sec. 3). (e) Mean average precision (MAP).

(and thus the rows) of M are smooth on the graph and enjoy an appealing "harmonic" structure [25] that was shown to respect the graph topology (we will elaborate on this in Sec. 3). It is thus interesting to look into M when α is small. To see how M behaves in such case, below we decompose it into a ranking matrix plus a constant matrix.

Denote by $\bar{L} = \Lambda^{-\frac{1}{2}} L \Lambda^{-\frac{1}{2}}$. It is easy to see that $\bar{L}$ is symmetric and positive semi-definite, and has the same rank $n-1$ as L (since the graph is connected). Let $\bar{L} = U \Gamma U^\top$ be the eigen-decomposition of $\bar{L}$ with eigenvalues $0 = \gamma_1 < \gamma_2 \leq \cdots \leq \gamma_n$. $U = (\mathbf{u}_1, \cdots, \mathbf{u}_n)$ is the orthonormal eigenvector matrix, with $\mathbf{u}_1 = (\frac{\sqrt{\lambda_1}}{\sqrt{\sum_i \lambda_i}}, \cdots, \frac{\sqrt{\lambda_n}}{\sqrt{\sum_i \lambda_i}})^\top$. Denote by $\bar{L}^\dagger$ the pseudo-inverse of $\bar{L}$, and denote by $\mathbf{1}$ the vector of all ones. We are ready to decompose M.

Theorem 2.1. *$M = C + E$, where $C = \frac{1}{\alpha \sum_i \lambda_i} \mathbf{1}\mathbf{1}^\top$, and $E = \Lambda^{-\frac{1}{2}} \left(\sum_{i=2}^{n} \frac{1}{\gamma_i + \alpha} \mathbf{u}_i \mathbf{u}_i^\top \right) \Lambda^{-\frac{1}{2}}$.*

As mentioned above, the cases with small α are particularly interesting as such metrics can be expected to respect the graph topology. It is thus intriguing to look into the limit of the ranking matrix E as $\alpha \to 0$.

Corollary 2.2. $\lim_{\alpha \to 0} E = \Lambda^{-\frac{1}{2}} \bar{L}^\dagger \Lambda^{-\frac{1}{2}}$.

By Theorem 2.1, when α is small, M is dominated by C. However, as C is constant, it has no effect on the ranking based on M. In other words, ranking by M is completely determined by E, which, by Corollary 2.2, converges to the pseudo-inverse of $\bar{L}$ doubly normalized by $\Lambda^{-\frac{1}{2}}$. Unlike personalized PageRank [20] and manifold ranking [29], which converge to limits solely determined by vertex degrees, the limit of E is meaningful by taking into account the graph structure. This property is desirable since it saves the trouble of selecting α to avoid biases. In the following, we highlight two special designs of Λ which relate to well-known metrics and are proven to be successful both theoretically and empirically.

Regularizer I. For $\Lambda = I$, we have $\bar{L} = L$, and by Corollary 2.2, $\lim_{\alpha \to 0} E = L^\dagger$, where $L^\dagger$ is the pseudo-inverse of the graph Laplacian. This implies that if α is sufficiently small, ranking by M is essentially the same as ranking by the pseudo-inverse of the graph Laplacian, which is widely used in clustering and recommendation [6] and proven to respect the graph structure [25].

Regularizer D. For $\Lambda = D$, we have $\bar{L} = L_{sym}$, and by Corollary 2.2, $\lim_{\alpha \to 0} E = D^{-\frac{1}{2}} L_{sym}^\dagger D^{-\frac{1}{2}}$. Let us consider the hitting times h_{ij} from every vertex i to hit a given vertex j. Then ranking by $(h_{ij})_{i=1,\ldots,n}$ is equivalent as ranking by the j-th column of $D^{-\frac{1}{2}} L_{sym}^\dagger D^{-\frac{1}{2}}$ (see supplement). This implies that if α is sufficiently small, ranking by the j-th column of M is essentially the same as ranking by the hitting times $(h_{ij})_{i=1,\ldots,n}$, which is a popular metric in many fields and shown to be meaningful [25]. Note that the columns of $M = (L + \alpha D)^{-1}$ also correspond to the degree normalized personalized PageRank [1], a popular local clustering method with theoretical guarantees.

2.2. Impact of Regularizer I and D

In this subsection, we explore the role of different regularizer empirically. To this end, we use a synthetic dataset shown in Fig. 3(a) which consists of two 20-dimensional Gaussians with different variances. Rankings by I and D are visualized in Fig. 3(b-c, f-g), and the mean average precisions (MAP) by taking each vertex as query, are summarized in Fig. 3(e). We can draw several interesting observations. First, for queries from the dense Gaussian (the

one with smaller variance), I performs much better than D, while for queries from the sparse Gaussian (the one with larger variance), D is extremely superior compared to I. Second, I retrieves the dense Gaussian almost perfectly, but behaves poorly on the sparse one. Finally, and probably more surprisingly, D works better on the sparse Gaussian than on the dense one, which is somewhat counterintuitive since a denser cluster is supposed to be easier to extract. Below we offer a probabilistic interpretation for these interesting behaviors, and defer rigorous arguments until Sec. 3.

2.2.1 A Probabilistic Perspective

What are the assumptions behind regularizer I and D? To understand this, we leverage a model called partially absorbing random walk (PARW) [26]. PARW is a random walk that at each step, it gets absorbed at the current state i with probability p_{ii}, or moves to its neighbor j with probability $(1-p_{ii}) \times \frac{w_{ij}}{d_i}$, where $p_{ii} = \frac{\alpha\lambda_i}{\alpha\lambda_i + d_i}$. The absorption probability a_{ij} that PARW starts from state i and gets absorbed at state j within finite steps can be derived in closed form, $A = [a_{ij}] \in \mathbb{R}^{n\times n} = (L+\alpha\Lambda)^{-1}\alpha\Lambda$. We can immediately see that for a query j, ranking by the j-th column of $M = (L+\Lambda)^{-1}$ is the same as ranking by the j-th column of A, i.e., the absorption probabilities of PARW getting absorbed at state j. We can gain some intuitions using absorption probabilities.

For $\Lambda = I$, $p_{ii} = \frac{\alpha}{\alpha + d_i}$, which is inversely proportional to the degree of i when $\alpha \to 0$. This means PARW has much larger mobility in the dense cluster than in the sparse clusters (the smaller p_{ii}, the larger the mobility of PARW at i). Therefore if the query is in the dense cluster, PARW from the dense cluster can easily hit the query, but PARW from the sparse cluster can hardly get to the query as it will be absorbed in the sparse cluster with high probability. Similarly, if the query is in the sparse cluster, the large absorption will prevent PARW from the same cluster to reach the query. This explains why I prefers the dense cluster.

For $\Lambda = D$, $p_{ii} = \frac{\alpha d_i}{\alpha d_i + d_i} = \frac{\alpha}{1+\alpha}$, which is constant at each state. This means PARW has the same mobility within each cluster. However, since the connection within the sparse cluster is weaker than it is in the dense cluster, PARW from the sparse cluster can go to the dense cluster more easily than the other way around. Therefore if the query is in the sparse cluster, PARW from the dense cluster will hardly reach it. Similarly, if the query is in the dense cluster, PARW from the sparse cluster is likely to hit it. This explains why D prefers the sparse cluster.

3. Analysis

In this section, we aim to quantify the impact of different regularizer Λ, which allows us to compare different Λ on the same graph. In particular, our intuitions about $\Lambda = I$ and $\Lambda = D$ will be made precise, and moreover, we propose a new Λ to combine their strengths.

3.1. Local Divergence

To quantify the impact of a regularizer, we introduce a notion of divergence to compute its impact on vertex subsets of the graph. It was shown in [25] that when α is small, the columns of the absorption probability matrix $A = [a_{ij}] = (L+\alpha\Lambda)^{-1}\alpha\Lambda$ (Sec. 2.2.1), if taken as functions on the underlying graph, enjoy a desirable harmonic structure which respects the graph topology. Since the j-th $(\forall j)$ column vector of the metric $M = (L+\alpha\Lambda)^{-1}$ is proportional to the j-th column vector of A, it enjoys the same structure as well.

W.l.o.g., let us take vertex 1 as the query. Denote by $\mathfrak{f}$ the first column vector of M and assume the vertices are sorted such that $\mathfrak{f}_1 > \mathfrak{f}_2 \geq \cdots \geq \mathfrak{f}_n$. The harmonic structure means that the ranking score of a vertex is approximately the weighted average of its neighbors, i.e., $\mathfrak{f}_i \approx \sum_{j\sim i} \frac{w_{ij}}{d_i}\mathfrak{f}_j, \forall i$. Our goal is to characterize the variation of $\mathfrak{f}$ with respect to Λ. To achieve this, we use a notion called "harmonic loss" proposed in [25], which is defined as:

$$\mathcal{L}_{\mathfrak{f}}(\mathcal{S}) := \sum_{i\in\mathcal{S}, j\in\bar{\mathcal{S}}} w_{ij}(\mathfrak{f}_i - \mathfrak{f}_j), \tag{2}$$

where $\mathcal{S} \subseteq \mathcal{V}$ is a subset of vertices. $\mathcal{L}_{\mathfrak{f}}(\mathcal{S})$ is referred to as the harmonic loss of the function $\mathfrak{f}$ on the subset $\mathcal{S}$. By the above definition, we can see that $\mathcal{L}_{\mathfrak{f}}(\mathcal{S})$ is actually the weighted gap of $\mathfrak{f}$ between any vertex set $\mathcal{S}$ and its complement $\bar{\mathcal{S}}$. Since in this paper, we only consider $\mathcal{L}_{\mathfrak{f}}$ on the local subsets given by $\mathcal{S}_k := \{1,\ldots,k\}$, we call $\mathcal{L}_{\mathfrak{f}}(\mathcal{S}_k)$ the *local divergence*[1] of $\mathfrak{f}$ on $\mathcal{S}_k$.

Let $\lambda(\mathcal{S}) = \sum_{i\in S}\lambda_i$. As shown below in Lemma 3.1, $\mathcal{L}_{\mathfrak{f}}(\mathcal{S}_k)$ can be represented by absorption probabilities (see Sec. 2.2.1), and strictly decreases as k increases. More importantly, its limit w.r.t. α only involves the regularizer Λ.

Lemma 3.1. (a) [25] $\mathcal{L}_{\mathfrak{f}}(\mathcal{S}_k) = \sum_{j\in\bar{\mathcal{S}}_k} a_{1j}$, (b) $\lim_{\alpha\to 0}\mathcal{L}_{\mathfrak{f}}(\mathcal{S}_k) = \lambda(\bar{\mathcal{S}}_k)/\lambda(\mathcal{V})$, $1 \leq k \leq n$.

W.l.o.g., denote by $\mathfrak{i}$ the first column vector of M with $\Lambda = I$, and assume that $\mathfrak{i}_1 > \mathfrak{i}_2 \geq \cdots \geq \mathfrak{i}_n$. Denote by $\mathfrak{d}$ the first column vector of M with $\Lambda = D$, and assume that $\mathfrak{d}_1 > \mathfrak{d}_2 \geq \cdots \geq \mathfrak{d}_n$. Let $d(\mathcal{S}) = \sum_{i\in S} d_i$. The following corollaries are obtained by applying $\Lambda = I$ and $\Lambda = D$ to Lemma 3.1(b).

Corollary 3.2. $\lim_{\alpha\to 0}\mathcal{L}_{\mathfrak{i}}(\mathcal{S}_k) = |\bar{\mathcal{S}}_k|/n$.

Corollary 3.3. $\lim_{\alpha\to 0}\mathcal{L}_{\mathfrak{d}}(\mathcal{S}_k) = d(\bar{\mathcal{S}}_k)/d(\mathcal{V})$.

[1] The term "divergence" is reminiscent to its counterpart in mathematics where it measures the magnitude of a vector field's source or sink at a given point [18].

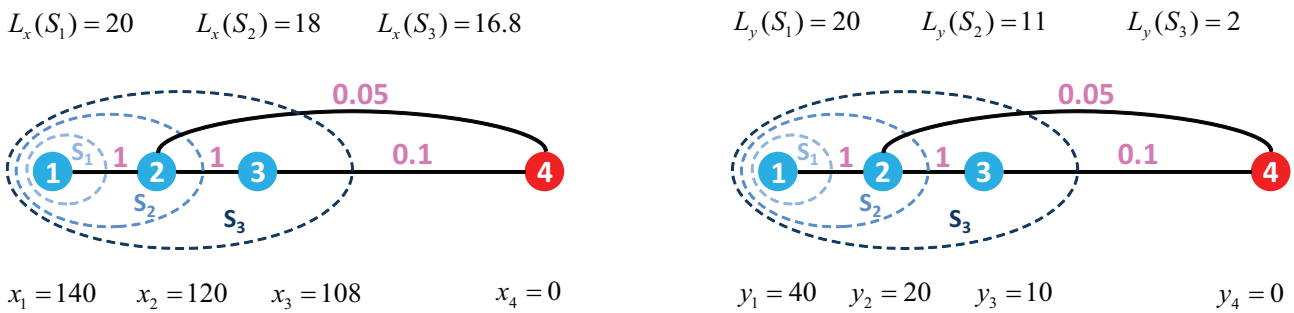


Figure 4. Functions x and y and their local divergence on a graph of two clusters $\{1,2,3\}$ and $\{4\}$ (edge weights indicated in pink).

3.2. Divergence Ratio

To understand and compare the impact of different regularizer Λ in retrieving the local cluster of a query, we develop an evaluation criterion in this subsection. Following last subsection, assume vertex 1 is the query and the cluster of interest $\mathcal{S}_c = \{1,\dots,c\}$, and suppose that $\mathfrak{f}$ ranks $\mathcal{S}_c$ on the top: $\mathfrak{f}_1 > \mathfrak{f}_2 \geq \cdots \geq \mathfrak{f}_c \geq \mathfrak{f}_{c+1} \geq \cdots \geq \mathfrak{f}_n$. To quantify the robustness of the ranking, it would make sense to consider the ratio of the gap between $\mathcal{S}_c$ and $\bar{\mathcal{S}}_c$ and the gap within $\mathcal{S}_c$, as follows:

$$r_{\mathfrak{f}}(\mathcal{S}_c) := \frac{\mathfrak{f}_c - \mathfrak{f}_{c+1}}{\sum_{k=1}^{c-1}(\mathfrak{f}_k - \mathfrak{f}_{k+1})} = \frac{\mathfrak{f}_c - \mathfrak{f}_{c+1}}{\mathfrak{f}_1 - \mathfrak{f}_c}. \tag{3}$$

Clearly, the larger the ratio is, the more robust the ranking is. However, without knowing the actual values of $\mathfrak{f}$, it seems rather difficult, if not impossible, to characterize $r_{\mathfrak{f}}(\mathcal{S}_c)$ on a general graph.

Recall that the local divergence of $\mathfrak{f}$ on $\mathcal{S}_k$ is defined as $\mathcal{L}_{\mathfrak{f}}(\mathcal{S}_k) = \sum_{i\in\mathcal{S}_k, j\in\bar{\mathcal{S}}_k} w_{ij}(\mathfrak{f}_i - \mathfrak{f}_j)$, which is exactly the weighted gap between $\mathcal{S}_k$ and $\bar{\mathcal{S}}_k$. This motivates us to characterize $\mathfrak{f}$ in terms of $\mathcal{L}_{\mathfrak{f}}(\mathcal{S}_k)$. Specifically, we may evaluate the robustness of $\mathfrak{f}$ in terms of its divergence ratio:

$$\mathcal{R}_{\mathfrak{f}}(\mathcal{S}_c) := \frac{\mathcal{L}_{\mathfrak{f}}(\mathcal{S}_c)}{\sum_{k=1}^{c-1}\mathcal{L}_{\mathfrak{f}}(\mathcal{S}_k)}. \tag{4}$$

Here, the numerator $\mathcal{L}_{\mathfrak{f}}(\mathcal{S}_c)$ is the weighted gap between the local cluster $\mathcal{S}_c$ and its complement $\bar{\mathcal{S}}_c$. For any $1 \leq k \leq c-1$, $\mathcal{L}_{\mathfrak{f}}(\mathcal{S}_k)$ breaks down to the weighted gap within $\mathcal{S}_c$ and the weighted gap between $\mathcal{S}_k$ and $\bar{\mathcal{S}}_c$, where the latter is part of $\mathcal{L}_{\mathfrak{f}}(\mathcal{S}_c)$ and not necessarily zero. The denominator is thus the combination of the accumulated weighted gap within $\mathcal{S}_c$ and some weighted gap between $\mathcal{S}_c$ and $\bar{\mathcal{S}}_c$. Since the connection between $\mathcal{S}_c$ and $\bar{\mathcal{S}}_c$ is much sparser than within $\mathcal{S}_c$, the divergence ratio $\mathcal{R}_{\mathfrak{f}}(\mathcal{S}_c)$ is dominated by the ratio of the weighted gap between $\mathcal{S}_c$ and $\bar{\mathcal{S}}_c$ and the accumulated weighted gap within $\mathcal{S}_c$, which appears to be a good surrogate for the gap ratio $r_{\mathfrak{f}}(\mathcal{S}_c)$. The following statement bounds the divergence ratio.

Theorem 3.4. $\mathcal{R}_{\mathfrak{f}}(\mathcal{S}_c) < 1/(c-1)$.

Also recall that we are only interested in the cases when α is small. We thus want to examine the limit of the divergence ratio $\mathcal{R}_{\mathfrak{f}}(\mathcal{S}_c)$, which, by Lemma 3.1(b), can be computed explicitly as follows:

$$\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{f}}(\mathcal{S}_c) := \frac{\lim_{\alpha\to 0}\mathcal{L}_{\mathfrak{f}}(\mathcal{S}_c)}{\sum_{k=1}^{c-1}\lim_{\alpha\to 0}\mathcal{L}_{\mathfrak{f}}(\mathcal{S}_k)} = \frac{\lambda(\bar{\mathcal{S}}_c)}{\sum_{k=1}^{c-1}\lambda(\bar{\mathcal{S}}_k)}. \tag{5}$$

Thanks to Eq. (5), we are able to compare the robustness of different $\mathfrak{f}$ (with different Λ) on a local cluster $\mathcal{S}_c$ numerically. Before doing so, we use one example to illustrate that the divergence ratio of a function reflects its gap ratio.

Example. Fig. 4 shows two ranking functions x and y on a graph with two clusters, along with their local divergence. $\mathcal{L}_x(\mathcal{S}_3)$ is much larger than $\mathcal{L}_y(\mathcal{S}_3)$, so is $x_3 - x_4$ to $y_3 - y_4$. Similarly, $\mathcal{L}_x(\mathcal{S}_1)+\mathcal{L}_x(\mathcal{S}_2)$ is larger than $\mathcal{L}_y(\mathcal{S}_1)+\mathcal{L}_y(\mathcal{S}_2)$, so is $x_1 - x_3$ to $y_1 - y_3$. Overall, $\mathcal{R}_x(\mathcal{S}_3) = \frac{16.8}{38}$ is much larger than $\mathcal{R}_y(\mathcal{S}_3) = \frac{2}{31}$. Correspondingly, $r_x(\mathcal{S}_3) = \frac{108}{32}$ is much larger than $r_y(\mathcal{S}_3) = \frac{10}{30}$. Note that if we remove the edge between vertices 2 and 4, it becomes a chain graph, and $\mathcal{R}_{\mathfrak{f}}$ is essentially the same as $r_{\mathfrak{f}}$ on a chain graph.

For $\mathcal{R}_{\mathfrak{f}}$ to be large, $\frac{\mathcal{L}_{\mathfrak{f}}(\mathcal{S}_c)}{\mathcal{L}_{\mathfrak{f}}(\mathcal{S}_k)}$ should be as large as possible for every $1 \leq k \leq c-1$. This means that for $\mathfrak{f}$ to be robust on $\mathcal{S}_c$, its local divergence $\mathcal{L}_{\mathfrak{f}}(\mathcal{S}_k)$ should drop as slowly as possible on $\mathcal{S}_k$. From Fig. 4, we can see that $\mathcal{L}_x(\mathcal{S}_1) = \mathcal{L}_y(\mathcal{S}_1)$, but $\mathcal{L}_x$ drops much slower than $\mathcal{L}_y$, resulting in a much more robust ranking function x than y.

3.3. Behaviors of Regularizer I and D

In this subsection, we analyze regularizer I and D and justify their behaviors observed in Sec. 2.2. By Corollaries 3.2 and 3.3, together with Eq. (5), the limiting divergence ratios of $\mathfrak{i}$ ($\Lambda = I$) and $\mathfrak{d}$ ($\Lambda = D$) are as follows:

$$\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{i}}(\mathcal{S}_c) = \frac{|\bar{\mathcal{S}}_c|}{\sum_{k=1}^{c-1}|\bar{\mathcal{S}}_k|}, \tag{6}$$

$$\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{d}}(\mathcal{S}_c) = \frac{d(\bar{\mathcal{S}}_c)}{\sum_{k=1}^{c-1}d(\bar{\mathcal{S}}_k)}. \tag{7}$$

With Eqs. (6) and (7), the following theorem compares the robustness of I and D on the same local cluster $\mathcal{S}_c$.

Theorem 3.5. (a) *If $d_i = b$, $\forall i$, for some constant b, then* $\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{i}}(\mathcal{S}_c) = \lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{d}}(\mathcal{S}_c)$.
(b) *Suppose for* $1 \le k < c$, $\frac{d(\mathcal{S}_c\setminus\mathcal{S}_k)}{|\mathcal{S}_c\setminus\mathcal{S}_k|} > \frac{d(\bar{\mathcal{S}}_c)}{|\bar{\mathcal{S}}_c|}$. *Then* $\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{i}}(\mathcal{S}_c) > \lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{d}}(\mathcal{S}_c)$.
(c) *Suppose for* $1 \le k < c$, $\frac{d(\mathcal{S}_c\setminus\mathcal{S}_k)}{|\mathcal{S}_c\setminus\mathcal{S}_k|} < \frac{d(\bar{\mathcal{S}}_c)}{|\bar{\mathcal{S}}_c|}$. *Then* $\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{i}}(\mathcal{S}_c) < \lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{d}}(\mathcal{S}_c)$.

Theorem 3.5 explains our first observation about I and D in Sec. 2.2. When the density of the graph is the same everywhere, i.e., every vertex in the graph has the same degree, I is essentially the same as D. However, when the density of the local cluster $\mathcal{S}_c$ is larger than its surrounding clusters, by Theorem 3.5(b), we have $\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{i}}(\mathcal{S}_c) > \lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{o}}(\mathcal{S}_c)$. Note that for $1 \le k < c$, $\frac{d(\mathcal{S}_c\setminus\mathcal{S}_k)}{|\mathcal{S}_c\setminus\mathcal{S}_k|} > \frac{d(\bar{\mathcal{S}}_c)}{|\bar{\mathcal{S}}_c|}$ means that the average degree of vertices within $\mathcal{S}_c$ is larger than that of outside $\mathcal{S}_c$, i.e., $\mathcal{S}_c$ is denser than $\bar{\mathcal{S}}_c$. Therefore, if α is sufficiently small, we have $\mathcal{R}_{\mathfrak{i}}(\mathcal{S}_c) > \mathcal{R}_{\mathfrak{d}}(\mathcal{S}_c)$, i.e., $\mathfrak{i}$ is more robust than $\mathfrak{d}$ on $\mathcal{S}_c$. This justifies the observation that I performs better than D on dense clusters. Similarly, by Theorem 3.5(c), we prove that D performs better than I on sparse clusters.

To explain our second observation that I prefers dense to sparse clusters, simply notice that by Eq. (6) $\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{i}}(\mathcal{S}_c)$ is the same regardless the density of $\mathcal{S}_c$, but extracting a sparse cluster requires much more "robustness" than extracting a dense cluster. In order to interpret our third observation that D prefers sparse to dense clusters, we characterize in the following lemmas the variation of $\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{d}}(\mathcal{S}_c)$ w.r.t. the density of $\mathcal{S}_c$.

Lemma 3.6. $\lim_{d(\mathcal{S}_c)/d(\bar{\mathcal{S}}_c)\to 0}\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{d}}(\mathcal{S}_c) = \frac{1}{c-1}$.

Lemma 3.7. $\lim_{d(\mathcal{S}_c)/d(\bar{\mathcal{S}}_c)\to \infty}\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{d}}(\mathcal{S}_c) = 0$, *if* $d_1 < td(\mathcal{S}_c)$ *for a fixed scalar* t, $0 < t < 1$.

By Lemma 3.6, when the local cluster $\mathcal{S}_c$ is extremely sparse, $\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{d}}(\mathcal{S}_c)$ reaches its upper bound $1/(c-1)$ (Theorem 3.4). This shows that $\mathfrak{d}$ is perfectly robust on $\mathcal{S}_c$. However, by Lemma 3.7, when the local cluster $\mathcal{S}_c$ is super dense, $\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{d}}(\mathcal{S}_c)$ reaches its lower bound 0. This shows that $\mathfrak{d}$ is not robust at all on $\mathcal{S}_c$. These results explain the behavior of D that it prefers sparse to dense clusters.

3.4. The Proposed Regularizer H

We have shown that regularizer I and D behave complementarily, and each has its own strength and weakness. Ideally, if the query is from a dense cluster, then I should be used; and if it is from a sparse cluster, then D would be a better choice. However, in practice, there is no reliable way to tell whether a query is in a "sparse" or "dense" cluster. This poses an interesting question: does it exist a regularizer that works for both dense and sparse clusters? If so, what is it? In this subsection, we address this practical issue by designing a new regularizer Λ which can automatically switch between the I mode and the D mode.

Our solution is inspired by the observations as follows. First, since on sparse clusters where the vertices are of relatively low degrees, I tends to fail while D works much better, it suggests that the "ideal" regularizer (λ_i's) for low degree vertices should be set relatively small (e.g., following D). Second, D does not perform well on dense clusters, which suggests that the regularizer for high degree vertices should not be set too large. Third, I works well on dense clusters, implying that a constant regularizer on high degree vertices may be desired. Combining these arguments, we propose to take $\Lambda = H = \mathrm{diag}(h_1, h_2, \cdots, h_n)$ with

$$h_i = \min(\hat{d}, d_i),\ i = 1, \ldots, n, \tag{8}$$

where $\hat{d}$ is the τ-th largest entry in $(d_1, d_2, \cdots, d_n)$ (e.g., the median), and w.l.o.g., we assume it is unique. One can see that H is essentially a mix of I and D – it equals to D at vertices with degree smaller than $\hat{d}$, and stays constant otherwise. Let $\mathfrak{h}$ denote the first column vector of $M = (L + \alpha H)^{-1}$, and w.l.o.g., assume $\mathfrak{h}$ ranks the local cluster $\mathcal{S}_c = \{1, \ldots, c\}$ on the top (vertex 1 being the query), i.e., $\mathfrak{h}_1 > \mathfrak{h}_2 \ge \cdots \ge \mathfrak{h}_c \ge \mathfrak{h}_{c+1} \ge \cdots \ge \mathfrak{h}_n$. We justify this choice of H below.

3.4.1 Justification

We first show that H behaves like D when the local cluster $\mathcal{S}_c$ is sparse. Assume that $\max_{i\in\mathcal{S}_c} d_i < \hat{d}$, and let $\mathcal{S}' = \{i | d_i < \hat{d}\}$. By Eq. (5) and the definition of H in Eq. (8), the limiting divergence ratio of $\mathfrak{h}$ can be written as follows:

$$\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{h}}(\mathcal{S}_c) = \frac{d(\mathcal{S}'\setminus\mathcal{S}_c) + \tau\hat{d}}{\sum_{k=1}^{c-1}\left(d(\mathcal{S}'\setminus\mathcal{S}_k) + \tau\hat{d}\right)}. \tag{9}$$

With Eq. (9) and Eq. (6), we are ready to compare the robustness of H and I on the same local cluster $\mathcal{S}_c$, as shown by the following theorem.

Theorem 3.8. *Suppose for* $1 \le k < c$, $\frac{d(\mathcal{S}_c\setminus\mathcal{S}_k)}{|\mathcal{S}_c\setminus\mathcal{S}_k|} < \frac{d(\mathcal{S}'\setminus\mathcal{S}_c)+\tau\hat{d}}{|\mathcal{S}_c|}$. *Then* $\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{i}}(\mathcal{S}_c) < \lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{h}}(\mathcal{S}_c)$.

By Theorem 3.8, if the density of the local cluster $\mathcal{S}_c$ is sparse to an extent, H is more robust than I. Indeed, by Lemma 3.9 below, when $\mathcal{S}_c$ is extremely sparse, $\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{h}}(\mathcal{S}_c)$ reaches its upper bound, which is exactly like D. This proves that by setting $h_i = d_i$ for low degree vertices, H behaves very similarly as D on sparse clusters.

Lemma 3.9. $\lim_{\max_{i\in\mathcal{S}_c} d_i/\hat{d}\to 0}\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{h}}(\mathcal{S}_c) = \frac{1}{c-1}$.

We next show that H behaves like I when the local cluster $\mathcal{S}_c$ is dense. Assume $\min_{i\in\mathcal{S}_c} d_i > \hat{d}$, and let $\mathcal{S}^* = \{i | d_i > \hat{d}\}$. By Eq. (5) and the definition of H in Eq. (8), the limiting divergence ratio of $\mathfrak{h}$ is as follows:

$$\lim_{\alpha\to 0} \mathcal{R}_{\mathfrak{h}}(\mathcal{S}_c) = \frac{|\mathcal{S}^* \setminus \mathcal{S}_c|\hat{d} + d(\bar{\mathcal{S}}^*)}{\sum_{k=1}^{c-1}\left(|\mathcal{S}^* \setminus \mathcal{S}_k|\hat{d} + d(\bar{\mathcal{S}}^*)\right)}. \quad (10)$$

By Eq. (10) and Eq. (7), we can compare the robustness of H and D on the same local cluster $\mathcal{S}_c$, as stated in the following theorem.

Theorem 3.10. *Suppose for* $1 \le k < c$, $\frac{d(\mathcal{S}_c\setminus\mathcal{S}_k)}{|\mathcal{S}_c\setminus\mathcal{S}_k|} > \frac{d(\bar{\mathcal{S}}_c)}{|\mathcal{S}^*\setminus\mathcal{S}_c|+d(\bar{\mathcal{S}}_*)/\hat{d}}$. *Then* $\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{h}}(\mathcal{S}_c) > \lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{d}}(\mathcal{S}_c)$.

By Theorem 3.10, if the density of the local cluster $\mathcal{S}_c$ is dense to an extent, H is more robust than D. In fact, by Eq. (10), we can see that $\lim_{\alpha\to 0}\mathcal{R}_{\mathfrak{h}}(\mathcal{S}_c)$ does not depend on the density within $\mathcal{S}_c$, which is exactly like I. This shows that by setting $\lambda_i = \hat{d}$ for high degree vertices, H behaves much like I on dense clusters.

Remark on $\hat{d}$: The performance of H depends on the choice of $\hat{d}$, and $\hat{d}$ should not be set too large or too small. If $\hat{d}$ is too large, H behaves much like D and there is no sufficient regularization for dense data. On the other hand, if $\hat{d}$ is too small, H behaves much like I and cannot do well on sparse data. By our empirical studies, we find that $\hat{d} = \text{median}(d_1, d_2, \cdots, d_n)$ works quite well in practice. We adopt this setting throughout our experiments.

The superiority of H can be immediately seen on the Two Gaussians example in Fig. 3, where H demonstrates a nice balance between I and D. On the dense Gaussian, H is very close to I and much better than D; while on the sparse Gaussian, it is very close to D and much better than I. Overall, H performs superior than either I or D. More evaluations on real datasets are reported in the next section.

4. Experimental Results

In this section, we compare regularizer I, D, and H for image retrieval on three large benchmark datasets: USPS, MNIST, and CIFAR-10.

Parameter Setup. We construct a weighted 20-NN graph for each dataset, including the Two Gaussians in Fig. 3. The edge weight between vertices i and j is set as $w_{ij} = \exp(-d_{ij}^2/\sigma)$ if i is within j's 20 nearest neighbors or vice versa, otherwise $w_{ij} = 0$, where d_{ij} is the Euclidean distance between vertices i and j. We set $\sigma = 0.2 \times s$ with s being the average square distance among each vertex to its 20-th nearest neighbor. For I, D, and H, we use the same α and set it to a very small number $\alpha = 1e-6$, to approximate the limiting case considered in this paper. For each dataset, we compute the mean average precision (MAP) on each class and on the entire dataset (the average of the MAP on all classes).

USPS Dataset. USPS[2] contains 9298 images of handwritten digits from 0 to 9 of size 16×16, with 1553, 1269, 929, 824, 852, 716, 834, 792, 708, and 821 in each class. We use each instance as query on the entire dataset.

MNIST Dataset. MNIST[3] [12] contains 70,000 images of handwritten digits from 0 to 9 of size 28×28, with 6903, 7877, 6990, 7141, 6824, 6313, 6876, 7293, 6825 and 6958 in each class. It consists of a training set of 60,000 examples and a test set of 10,000 examples. We use each instance in the test set as query (and all 70,000 images as the database).

CIFAR-10 Dataset. CIFAR-10[4] consists of 60,000 tiny color images of size 32x32 in 10 mutually exclusive classes, with 6,000 in each class. There are 50,000 training images and 10,000 test images. Each image is represented by a 512-dimensional GIST feature vector [19]. We use each test image as query (and all 60,000 images as the database).

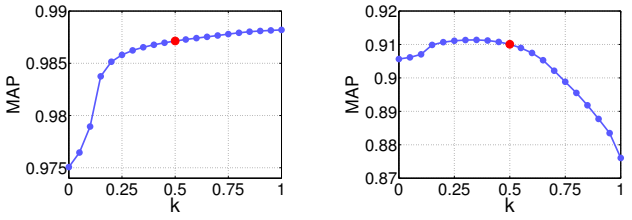
The results are shown in Table 1, where $\hat{d}$ denotes the median degree in each class and on the entire graph. For each class, we highlight the results when H is biased to the significantly better regularizer. We can draw several observations. First, by $\hat{d}$, we can see that it is common that real image clusters are of varied density. Some classes can be highly dense because images of that class have more similar features, e.g., digit "1" in USPS and MNIST, "plane" and "ship" in CIFAR; while some can be rather sparse due to less similar features, e.g., digit "2" in USPS and MNIST, "dog" and "cat" in CIFAR. Second, I and D show distinctive yet complementary behaviors. For example, I is much better than D on dense classes, e.g., "plane" and "ship" in CIFAR, and digits "1" and "7" in MNIST. In contrast, D performs much better than I on sparse classes, e.g., digits "2", "4", and "5" in USPS, and "auto", "cat", "dog", "horse", and "truck" in CIFAR. Third, our proposed H successfully adapts to the data density and combines the strengths of I and D (as highlighted), thus achieving the best overall retrieval results (last column in Table 1) on all the three datasets. We also include the results of personalized PageRank (PR) [20] and manifold ranking (MR) [29] for comparison, using the parameters suggested in their original papers. H is significantly better than both methods on almost every cluster of each dataset.

To see how H performs w.r.t. $\hat{d}$, we test H with different $\hat{d}$ on the entire USPS dataset, using each image from the densest cluster (digit "1") and the sparsest cluster (digit "2") as query, respectively. The results are shown in Fig. 5. We run through H with $\tau = \lfloor n \times k \rfloor$, where $k = [0 : 0.05 : 1]$ (let $\tau = 1$ when $k = 0$). Note that when $\tau = 1$, $\hat{d}$ is equal to the largest vertex degree, and H is essentially the same as D; and when $\tau = n$, $\hat{d}$ is equal to the smallest vertex degree, and H is essentially the same as I. In Fig. 5, H

[2] http://www-stat.stanford.edu/ tibs/ElemStatLearn/
[3] http://yann.lecun.com/exdb/mnist/
[4] http://www.cs.toronto.edu/ kriz/cifar.html

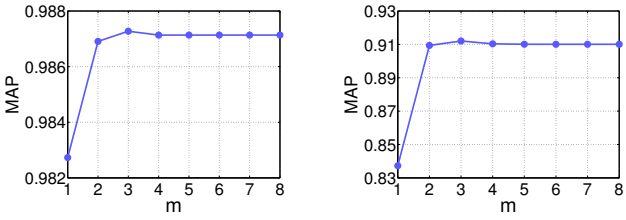
Table 1. Mean average precision on the USPS, MNIST, and CIFAR datasets.

		0	1	2	3	4	5	6	7	8	9	All
USPS	$\hat{d}$	0.76	18.66	0.04	0.13	0.27	0.05	0.83	1.68	0.17	1.70	0.47
	I	.9805	**.9882**	.8760	.8926	.6462	.7781	**.9401**	.9194	.7460	**.7296**	.8497
	D	.9819	.9751	**.9057**	.8926	**.6816**	**.7972**	.9231	.9153	.7450	.6959	.8514
	H	.9797	**.9871**	**.9101**	.8961	**.6819**	**.7971**	**.9408**	.9167	.7679	**.7231**	**.8601**
	PR	.8860	.9720	.6080	.7639	.4879	.5684	.8374	.8253	.6255	.7022	.7277
	MR	.9570	.9871	.8272	.8273	.4671	.6303	.9167	.8225	.6750	.7191	.7829
MNIST	$\hat{d}$	0.30	11.18	0.07	0.15	0.36	0.15	0.49	1.06	0.11	0.79	0.32
	I	.9877	**.9759**	.9269	.8867	.7916	.8004	.9745	**.8848**	.8118	.6602	.8700
	D	.9881	.9249	**.9324**	.8744	.8102	.8097	.9706	.8502	.8161	.6573	.8634
	H	.9868	**.9746**	**.9397**	.8831	.8002	.8070	.9742	**.8832**	.8341	.6613	**.8744**
	PR	.8867	.7444	.6574	.7006	.5941	.5750	.8303	.6916	.5874	.5916	.6859
	MR	.9803	.9436	.8897	.8166	.6355	.7152	.9546	.7883	.7140	.6463	.8084
CIFAR		plane	auto	bird	cat	deer	dog	frog	horse	ship	truck	All
	$\hat{d}$	0.65	0.15	0.33	0.13	0.36	0.15	0.27	0.16	0.51	0.16	0.23
	I	**.2999**	.2760	**.1570**	.1320	.1703	.1848	.2949	.2243	**.3195**	.2493	.2308
	D	.2387	**.3049**	.1454	**.1562**	.1581	**.2141**	.2901	**.2488**	.2835	**.2741**	.2314
	H	**.2917**	**.2945**	**.1552**	**.1496**	.1621	**.2054**	.2891	**.2342**	**.3128**	**.2609**	**.2356**
	PR	.2335	.2050	.1418	.1007	.2136	.1403	.2612	.1571	.2655	.1701	.1889
	MR	.2296	.1513	.1286	.0821	.1715	.1022	.1924	.1201	.2321	.1124	.1522



(a) (b)

Figure 5. Sensitivity of H w.r.t. $\hat{d}$ on the USPS dataset. (a) Queries from the densest cluster. (b) Queries from the sparsest cluster.



(a) (b)

Figure 6. Stability of H w.r.t. α on the USPS dataset. (a) Queries from the densest cluster. (b) Queries from the sparsest cluster.

with median degree ($k = 0.5$) is highlighted in red. We can immediately see that I ($k = 1$) and D ($k = 0$) show opposite performances on the two clusters. On digit "1" (dense), H with $k \geq 0.5$ is close to I ($k = 1$), while on digit "2" (sparse), H with $k \leq 0.5$ is close to D ($k = 0$), indicating that $k = 0.5$ achieves a nice balance between I and D. We can also observe that variations of H are slow around $k = 0.5$ on both clusters, which shows that H is stable if $\hat{d}$ is not far away from the median degree.

To test the stability of H when α is small, we run through H with $\alpha = 10^{-m}$, where $m = [1 : 1 : 8]$, on the entire USPS dataset. Again, we use each image from the densest cluster (digit 1) and the sparsest cluster (digit 2) as query, respectively. It can be seen that on either cluster, MAP of H increases when α decreases and becomes stable quickly, which confirms our analysis in Sec. 2.

5. Conclusion

The contributions of this paper can be summarized as follows. By investigating the fundamental design of Laplacian-based similarity metrics, we provide new insights into widely used metrics including hitting times and the pseudo-inverse of graph Laplacian. We establish rigorous analysis to justify our findings. We also propose a new metric to solve the model selection problem, which has been successfully applied in image retrieval. We expect more graph-based visual applications to benefit from this work.

Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable comments.

References

[1] R. Andersen, F. Chung, and K. Lang. Local graph partitioning using pagerank vectors. In *FOCS*, pages 475–486, 2006. 1, 3

[2] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS*, volume 14, pages 585–591, 2001. 1

[3] Y. Boykov and G. Funka-Lea. Graph cuts and efficient nd image segmentation. *International Journal of Computer Vision*, 70(2):109–131, 2006. 1

[4] F. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997. 2

[5] R. Fergus, Y. Weiss, and A. Torralba. Semi-supervised learning in gigantic image collections. In *NIPS*, pages 522–530, 2009. 1

[6] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 19(3):355–369, 2007. 1, 3

[7] L. Grady. Random walks for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11):1768–1783, 2006. 1

[8] J. He, M. Li, H.-J. Zhang, H. Tong, and C. Zhang. Manifold-ranking based image retrieval. In *ACM Multimedia*, pages 9–16, 2004. 1

[9] X. He, S. Yan, Y. Hu, P. Niyogi, and H.-J. Zhang. Face recognition using laplacianfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):328–340, 2005. 1

[10] G. Irie, Z. Li, X.-M. Wu, and S.-F. Chang. Locally linear hashing for extracting non-linear manifolds. In *CVPR*, pages 2123–2130, 2014. 1

[11] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Computer Science Department, University of Toronto, Tech. Rep*, 2009. 2

[12] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 7

[13] K. Lee, J. Ho, and D. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 27(5):684–698, 2005. 2

[14] Z. Li, J. Liu, and X. Tang. Pairwise constraint propagation by semidefinite programming for semi-supervised classification. In *ICML*, pages 576–583, 2008. 1

[15] Z. Li, X.-M. Wu, and S.-F. Chang. Segmentation using superpixels: A bipartite graph partitioning approach. In *CVPR*, pages 789–796, 2012. 1

[16] W. Liu, J. Wang, S. Kumar, and S.-F. Chang. Hashing with graphs. In *ICML*, pages 1–8, 2011. 1

[17] L. Lovász. Random walks on graphs: A survey. *Combinatorics, Paul erdos is eighty*, 2(1):1–46, 1993. 1

[18] J. E. Marsden and A. Tromba. *Vector calculus*. Macmillan, 2003. 4

[19] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001. 7

[20] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. 1999. 1, 3, 7

[21] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000. 1

[22] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2579-2605):85, 2008. 1

[23] U. von Luxburg, A. Radl, and M. Hein. Hitting and commute times in large random neighborhood graphs. *Journal of Machine Learning Research*, 15:1751–1798, 2014. 1

[24] K. Weinberger and L. K. Saul. Unsupervised learning of image manifolds by semidefinite programming. In *CVPR*, volume 2, pages 988–995, 2004. 1

[25] X.-M. Wu, Z. Li, and S.-F. Chang. Analyzing the harmonic structure in graph-based learning. In *NIPS*, 2013. 1, 3, 4

[26] X.-M. Wu, Z. Li, A. M.-C. So, J. Wright, and S.-F. Chang. Learning with partially absorbing random walks. In *NIPS*, 2012. 4

[27] X.-M. Wu, A. M.-C. So, Z. Li, and R. S.-Y. Li. Fast graph laplacian regularized kernel learning via semidefinite–quadratic–linear programming. In *NIPS*, pages 1964–1972, 2009. 1

[28] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *NIPS*, pages 1601–1608, 2004. 1

[29] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In *NIPS*, 2004. 1, 3, 7