Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence

Investigating the Effectiveness of Laplacian-Based Kernels in Hub Reduction

Ikumi Suzuki
Nara Institute of Science& Technology
Ikoma, Nara - Japan
ikumi-s@is.naist.jp

Kazuo Hara
National Institute of Genetics
Mishima, Shizuoka - Japan
kazuo.hara@gmail.com

Masashi Shimbo
Nara Institute of Science& Technology
Ikoma, Nara - Japan
shimbo@is.naist.jp

Yuji Matsumoto
Nara Institute of Science& Technology
Ikoma, Nara - Japan
matsu@is.naist.jp

Marco Saerens
Université Catholique de Louvain
Louvain-la-Neuve - Belgium
marco.saerens@uclouvain.be

Abstract

A "hub" is an object closely surrounded by, or very similar to, many other objects in the dataset. Recent studies by Radovanović et al. indicate that in high dimensional spaces, hubs almost always emerge, and objects close to the data centroid tend to become hubs. In this paper, we show that the family of kernels based on the graph Laplacian makes all objects in the dataset equally similar to the centroid, and thus they are expected to make less hubs when used as a similarity measure. We investigate this hypothesis using both synthetic and real-world data. It turns out that these kernels suppress hubs in some cases but not always, and the results seem to be affected by the size of the data. However, for the datasets in which hubs are indeed reduced by the Laplacian-based kernels, these kernels work well in ranking and classification tasks. This result suggests that the amount of hubs, which can be readily computed in an unsupervised fashion, can be a yardstick of whether Laplacian-based kernels work effectively for a given data.

Introduction

In recent studies, Radovanović et al. investigated *hubs* that emerge in high dimensional space (Radovanović, Nanopoulos, and Ivanović 2010a; 2010b). A hub is an object similar (or close) to many other objects in a dataset. Radovanović et al. observed that hub objects emerge as dimension increases, for a number of common similarity or distance measures. They also made a notable finding that the objects closer (more similar) to the data mean, or *centroid*, tend to become hubs.

Hub objects emerge even in space of moderately high dimension (e.g., 50-dimensions), whereas systems for real data analysis, such as those for natural language processing, often deal with more than one million features (dimensions).

As Radovanović et al. have pointed out, hubs impair the accuracy of k-nearest neighbor ($k$nn) classification. In $k$nn classification, the label of a test object is predicted by the (weighted) majority voting of the $k$nn objects whose labels are known. If test objects follow the same distribution as that of the objects in the dataset, hubs in the dataset should frequently appear in the $k$nn list for test objects as well. As a result, hub objects pose strong bias on the predicted labels, causing the classification results to be inaccurate. As we will discuss in a later section, hubs also impair information retrieval, and the label propagation methods for semi-supervised classification.

In this paper, we examine if Laplacian-based kernels, such as the commute-time kernels (Saerens et al. 2004) and the regularized Laplacian (Chebotarev and Shamis 1997; Smola and Kondor 2003), are effective for reducing hubs. We explore Laplacian-based kernels based on our observation that in the implicit feature space, the inner product with the centroid is uniform for all objects in the dataset; thus, no objects are closer to the centroid. According to Radovanović et al., objects close to the centroid become hubs, and we expect these kernels are more robust to the hubness phenomenon. We empirically examine if Laplacian-based kernels reduce hubs and consequently improve the performance of information retrieval as well as multi-class and multi-label k-nearest neighbor classification.

Laplacian-based Kernels for Graph Vertices

We first present a brief review of Laplacian-based kernels.

Let $\mathcal{G}$ be an undirected graph with n vertices, and let $\mathbf{A}$ be its adjacency matrix. The edges of $\mathcal{G}$ may have positive weights representing the degree of similarity between vertices. In this case, $\mathbf{A}$ is an affinity matrix holding the edge weights as its components. The *(combinatorial) Laplacian* $\mathbf{L}$ of $\mathcal{G}$ is an $n \times n$ matrix defined as

$$\mathbf{L} = \mathbf{D} - \mathbf{A}, \tag{1}$$

where $\mathbf{D}$ is a diagonal matrix with diagonals $[\mathbf{D}]_{ii} = \sum_j [\mathbf{A}]_{ij}$. $\mathbf{L}$ is positive semidefinite and has n orthogonal eigenvectors $\mathbf{u}_i$ and n corresponding eigenvalues λ_i. We assume that the indices for eigenvalues/eigenvectors are arranged in ascending order of eigenvalues, $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$. A well-known property of $\mathbf{L}$ is that $\lambda_1 = 0$ and $\mathbf{u}_1 = \mathbf{1}$ (a vector of all 1's).

Laplacian-based Kernels

In machine learning community, graph Laplacian has been used as the building block of various kernels defining inner products between vertices. Below are the most popular of such Laplacian-based kernels.

Commute-time kernels (Saerens et al. 2004)

$$\mathbf{L}_{\mathrm{CT}} = \mathbf{L}^{+} \quad \text{(pseudo-inverse of } \mathbf{L}\text{)}, \tag{2}$$

Regularized Laplacian (Chebotarev and Shamis 1997; Smola and Kondor 2003)

$$\mathbf{L}_{\mathrm{RL}} = (\mathbf{I}+\beta\mathbf{L})^{-1}, \tag{3}$$

Diffusion kernels (Kondor and Lafferty 2002)

$$\mathbf{L}_{\mathrm{DF}} = \exp(-\beta\mathbf{L}), \tag{4}$$

where $\beta(\geq 0)$ is a parameter of the regularized Laplacian and the diffusion kernels. Note that while we do not discuss them in this paper, variations of these kernels exist which use the *normalized Laplacian* $\mathscr{L} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$ (Chung 1997) in place of $\mathbf{L}$ in their definition.

The above kernels can be interpreted as transformations of Laplacian $\mathbf{L}$ through eigenvalue regularization (Smola and Kondor 2003). To be precise, all the Laplacian-based kernels above (henceforth denoted by $\mathbf{K}$) can be decomposed as follows, using n pairs of eigenvalues and eigenvectors $\{(\lambda_i, \mathbf{u}_i)\}$ $(i = 1, \ldots, n)$ of $\mathbf{L}$.

$$\mathbf{K} = \sum_{i=1}^{n} r(\lambda_i)\mathbf{u}_i\mathbf{u}_i^{\mathrm{T}}, \tag{5}$$

where $r : [0,\infty) \to [0,\infty)$ is a *regularization operator*, which characterizes each Laplacian-based kernel. For the three kernels above,

Commute-time kernels $r(\lambda) = \begin{cases} 0, & \lambda = 0; \\ 1/\lambda & \lambda \neq 0, \end{cases}$

Regularized Laplacian $r(\lambda) = 1/(1+\beta\lambda)$,

Diffusion kernels $r(\lambda) = \exp(-\beta\lambda)$.

As Eq. (5) shows, Laplacian-based kernels have the same eigenvectors as Laplacian $\mathbf{L}$. Their eigenvalues, on the other hand, are transformed by function $r(\lambda)$. To suppress the contribution of large λ's which represent high-frequency components, $r(\lambda)$ is in general a non-increasing function for $\lambda > 0$.

In the rest of the paper, we focus on the commute-time kernels $\mathbf{L}_{\mathrm{CT}}$ and the regularized Laplacian $\mathbf{L}_{\mathrm{RL}}$. Diffusion kernels $\mathbf{L}_{\mathrm{DF}}$ show properties similar to $\mathbf{L}_{\mathrm{RL}}$.

Hubs in High Dimensional Space

High dimensionality causes various problems that go under the name of *curse of dimensionality*. The most well-known "curse" includes overfitting (Hastie, Tibshirani, and Friedman 2001; Bishop 2006) and distance concentration (Beyer et al. 1999; François, Wertz, and Verleysen 2007).

The "emergence of hubs" is a new type of the curse which has been discovered only recently (Radovanović, Nanopoulos, and Ivanović 2010a). This phenomenon particularly affects methods based on nearest neighbor search, i.e., those which list objects similar (or near) to a query object according to a certain similarity (or distance) measure. Because hub objects are similar to many other objects in the dataset, they appear in the nearest neighbor lists of those many objects, Thus, search results become less meaningful, as the same objects (hubs) are included in the search results irrespective of the query. And Radovanović et al. found that hubs nearly always appear in high dimensional data.

Applications affected by hubs include information retrieval and k-nearest neighbor classification. Graph-based semi-supervised classification methods, such as label propagation (Zhu, Ghahramani, and Lafferty 2003; Zhou et al. 2004), may also be affected, as these methods are typically run on k-nearest neighbor graphs.

Whether or not hubs exist in a dataset can be checked by counting the number of times that each object $\mathbf{x}$ appears in the k-nearest neighbor list of other objects. Let this number be $N_k(\mathbf{x})$. If hubs exist in the dataset, the distribution of N_k should skew to the right (provided that $k \ll n$, where n is the number of the objects).

In a manner similar to (Radovanović, Nanopoulos, and Ivanović 2010a), we illustrate the emergence of hubs using synthetic data consisting of 500 objects, each of which is a d-dimensional binary vector. To generate this data set, we first sample, for each dimension $i = 1, \ldots, d$, a real number from the log-normal distribution with mean 5 and variance 1, and compute its rounded integer n_i. We then choose n_i objects (vectors) out of 500 uniformly at random, and assign 1 to their ith component. After 500 d-dimensional binary vectors are generated in this way, we measure their pairwise similarity by the cosine of the angle between them.

The histograms of N_{10} frequency for two datasets with different dimensions d ($d = 10, 50$) are shown in the top panels of Figure 1. We can see objects with extremely large N_{10} values (e.g., the point at $N_{10} = 60$) in the top right panel (50-dimensional data), while no such points can be seen for 10-dimensional data.

Another important finding by Radovanović et al. is that in high dimensional spaces, objects similar (or close) to the data mean (centroid) tend to become hubs. We can verify this with the dataset of 50-dimensional vectors above. The bottom panels of Figure 1 are the scatter plots of N_{10} values of the data objects against their cosine similarity to the centroid. For $d = 50$ (high-dimensional data; bottom-right), N_{10} values show a strong correlation with the similarity to the centroid, whereas for $d = 10$ (low-dimensional data; bottom-left), the correlation is much weaker.

Hubness Phenomenon and Laplacian-based Kernels

If objects close to the data centroid tend to become hubs, a possible direction to their reduction should be to seek a similarity (or distance) measure which evaluates all objects equally similar to (or distant from) the centroid. We show that the Laplacian-based kernels indeed give measures which meet this requirement.

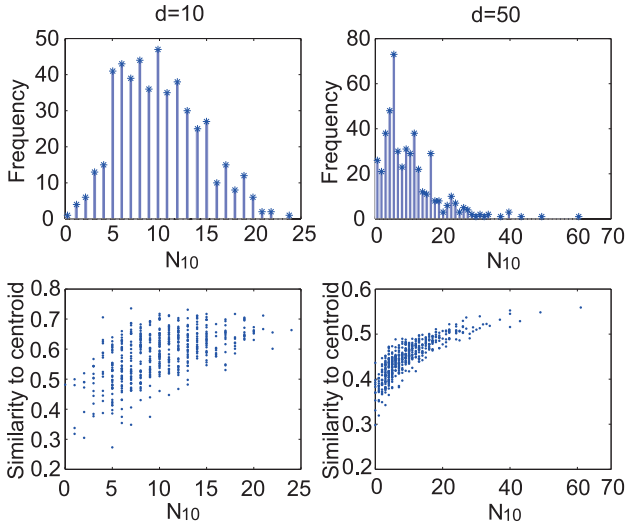


Figure 1: Top panels: Histograms of N_{10} frequency for two synthetic datasets in low ($d = 10$) and high ($d = 50$) dimensional feature spaces. Bottom panels: Scatter plots of the N_{10} value of an object against its similarity to the centroid. Each dot corresponds to a data object.

Centroid in the kernel-induced feature space

Suppose we have n data objects, $\mathscr{X} = \{\mathbf{x}_i\}$ $(i = 1, \ldots, n)$ in a vector space $\mathbb{D}$. We are also given a kernel $K : \mathbb{D} \times \mathbb{D} \mapsto \mathbb{R}$, which, for now, is not necessarily the Laplacian-based kernels introduced above.

Let $\mathbb{F}$ be the implicit feature space induced by kernel K, and $\phi(\cdot)$ be its associated feature mapping; i.e., a mapping of an object in $\mathbb{D}$ to its image in $\mathbb{F}$. Let $\mathbf{K}$ be the $n \times n$ Gram matrix of K computed for the dataset. Thus, component $[\mathbf{K}]_{ij}$ of matrix $\mathbf{K}$ is the inner product of $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$ in $\mathbb{F}$, or,

$$[\mathbf{K}]_{ij} = K(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle.$$

And the data centroid in the feature space $\mathbb{F}$, which we denote by $\overline{\phi}$, is given by

$$\overline{\phi} = \frac{1}{n} \sum_{i=1}^{n} \phi(\mathbf{x}_i).$$

Note that $\overline{\phi}$ differs from the data centroid in the original vector space $\mathbb{D}$, and, in general, from its image in $\mathbb{F}$, because $\phi(\cdot)$ can be non-linear.

Now the inner product between $\phi(\mathbf{x}_i)$ and the data centroid $\overline{\phi}$ in $\mathbb{F}$ is

$$\langle \phi(\mathbf{x}_i), \overline{\phi} \rangle = \langle \phi(\mathbf{x}_i), \frac{1}{n} \sum_{j=1}^{n} \phi(\mathbf{x}_j) \rangle = \frac{1}{n} \sum_{j=1}^{n} \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$$

$$= \frac{1}{n} \sum_{j=1}^{n} [\mathbf{K}]_{ij} = \frac{1}{n} [\mathbf{K}\mathbf{1}]_i. \tag{6}$$

Thus, it is the mean of the inner products between the ith object and all objects in the dataset, taken in the feature space induced by K. The last two equalities show that this quantity can be calculated simply by taking the mean of the ith row of the Gram matrix $\mathbf{K}$.

Laplacian-based kernels and similarity to the centroid

We now restrict K to Laplacian-based kernels, i.e., those which can be expressed as in Eq. (5). We show that these kernels define similarity measures which make the data centroid equally similar to all objects in the dataset.

Because Laplacian-based kernels assume that the data is represented as a graph, we treat the vector dataset $\mathscr{X}$ as a fully-connected graph. In this graph, data objects $\mathbf{x}_i$ corresponds to vertices, and edge weights are given by the pairwise similarity of objects measured in the original vector space $\mathbb{D}$.[1] In other words, the weighted adjacency matrix $\mathbf{A}$ of this fully-connected graph is given by the all-pairs similarity matrix for the dataset computed in $\mathbb{D}$. There may be many ways to measure similarity, but we only require that the similarity score given by $\mathbf{A}$ be non-negative and symmetric; hence $[\mathbf{A}]_{ij} = [\mathbf{A}]_{ji} \geq 0$ for all i, j. Given such an $\mathbf{A}$, we compute the graph Laplacian and then a Laplacian-based kernel matrix $\mathbf{K}$, e.g., using one of Eqs. (2)–(4).

Now, recall that the Laplacian-based kernels share the same eigenvectors as the Laplacian $\mathbf{L}$ from which they are computed, but the eigenvalues are transformed by $r(\cdot)$; see Eq. (5). In particular, for the smallest eigenvalue λ_1 of $\mathbf{L}$ and its corresponding eigenvector $\mathbf{u}_1$, it holds that $\mathbf{K}\mathbf{u}_1 = r(\lambda_1)\mathbf{u}_1$. And since $\mathbf{u}_1 = \mathbf{1}$ and $\lambda_1 = 0$, we have

$$\mathbf{K}\mathbf{1} = r(0)\mathbf{1}. \tag{7}$$

By Eq. (6), the left-hand side of this equation becomes

$$\mathbf{K}\mathbf{1} = n \begin{bmatrix} \langle \phi(\mathbf{x}_1), \overline{\phi} \rangle \\ \vdots \\ \langle \phi(\mathbf{x}_n), \overline{\phi} \rangle \end{bmatrix}. \tag{8}$$

On the other hand, the right-hand side of Eq. (7) is a constant vector whose components are all equal. It follows that all the components in Eq. (8) are equal. In other words,

$$\langle \phi(\mathbf{x}_1), \overline{\phi} \rangle = \langle \phi(\mathbf{x}_2), \overline{\phi} \rangle = \cdots = \langle \phi(\mathbf{x}_n), \overline{\phi} \rangle.$$

Thus, in the feature space induced by K, the inner products between the centroid and all object in the dataset are equal.

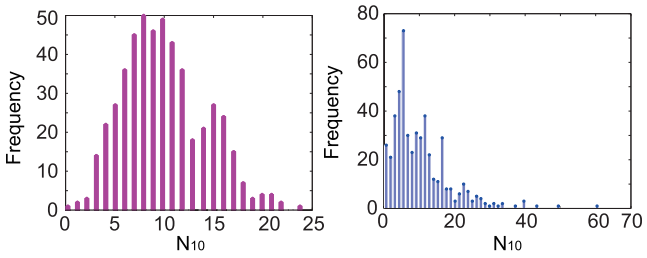
Remark The above property holds only if the components (inner products in the feature space) of Laplacian-based kernels $\mathbf{K}$ are used as they are as similarity scores. The similarity to the centroid may not be uniform if the closeness of objects is measured by distance in $\mathbb{F}$, i.e., via

$$d_{\mathbb{F}}(\mathbf{x}_i, \mathbf{x}_j) = ([\mathbf{K}]_{ii} + [\mathbf{K}]_{jj} - 2[\mathbf{K}]_{ij})^{1/2}. \tag{9}$$

We will show in later experiments that using distance in the feature space of Laplacian-based kernels in fact promotes hubs, and is always a bad idea.

According to Radovanović et al., objects close (or similar) to the centroid become hubs. As shown above, Laplacian-based kernels provide a similarity measure which makes data objects equally similar to the centroid. For this reason, we can expect them to suppress emergence of hubs.

[1] If a distance measure is given instead of similarity, we assume it is converted to a similarity in a standard way, e.g., by taking its reciprocal, or by using a Gaussian kernel.



(a) Commute-time kernel ($\mathbf{L}_{\mathrm{CT}}$) (b) Cosine similarity (Cos)

Figure 2: Histograms of N_{10} frequency for the synthetic 50-dimensional dataset of Figure 1: (a) commute-time kernel and (b) cosine similarity.

Experiments

We apply Laplacian-based kernels to real and synthetic datasets to see whether hubs are reduced by these kernels.

Synthetic data

First, as an illustration, we apply the commute-time kernels on the same 50-dimensional dataset we used previously to plot Figure 1. Figure 2(a) shows the histograms of N_{10} frequency for the commute-time kernel. For ease of comparison, (b) duplicates the top-right panel of Figure 1, which plots the histogram for cosine similarity. We see that with the commute-time kernel, no objects exhibit extremely large N_{10} values. Hence, the kernel has worked as expected for this dataset, and mitigated the hubness phenomenon.

Real data

For real data, we examine not only whether hubs are reduced by Laplacian-based kernels, but also whether they contribute to improved accuracy in tasks that use these datasets. We consider three tasks: (1) ranking (information retrieval), (2) multi-class classification, and (3) multi-label classification. These tasks are chosen because they require fine-grained similarity measures to distinguish individual data objects, which are not necessary for tasks such as binary classification.

Ranking task We rank biomedical terms in the MeSH thesaurus tree[2], to simulate mapping a new term onto the thesaurus.

To be precise, suppose we found a term in a text corpus, which is not present in an existing thesaurus. We want to register this new term into the thesaurus, but we first have to identify position in the thesaurus tree at which this "query" term must be inserted. To this end, we compare the similarity of the sentential context in which the query term appear, with the contexts in which existing terms in the thesaurus appear in a corpus. The intuition here is that the term must be placed near its synonyms, and synonyms often appear in similar contexts. Thus, we place the new (query) term near the existing terms in the thesaurus whose contextual similarity with the term is the largest.

[2]http://www.nlm.nih.gov/mesh/2009/introduction/introduction.html

To simulate this scenario, we treat each term in MeSH one by one as a query term, and see if its location in the MeSH thesaurus tree can be recovered with the above method; i.e., if the terms nearest to the query term in the thesaurus can be ranked higher than those located farther (in the MeSH tree).

Hence, for each term in MeSH, we rank other terms by the similarity of sentential contexts in which they appear, collected from abstracts in MEDLINE 2009[3]. The baseline measure evaluates the context similarity of terms by the cosine between "bag-of-words" feature vectors, which consist of the frequency of words occurring in the neighborhood of the terms in the corpus. We then compare this cosine similarity with the regularized Laplacian and commute-time kernels computed from the cosine similarity matrix.

In this task, a similarity measure is deemed better if it ranks terms located near the query term in the MeSH tree higher in the ranking for the query term. Because different query terms have different nearby terms in the MeSH tree, the similarity measure is required to return distinct rankings for each query term. If hub objects (terms) exist that tend to take higher positions in many rankings, they are likely to render the rankings more similar, and thus are harmful.

In this experiment, we make four datasets. These datasets consist of the set of terms under the top MeSH categories A, B, C, and D, respectively.

Multi-class classification For multi-class classification, we use two document classification datasets: Reuters-52[4], and TDT2-30[5]. A document in these datasets is classified into one of 52 and 30 categories, respectively. For Reuters-52, we used the default training-test data split accompanying the dataset. For TDT2-30, we randomly split the data into halves. For these tasks, we classify test documents by k-nearest neighbor ($k$nn) classification. The similarity measures used with $k$nn are the cosine between bag-of-words feature vectors of documents, and the regularized Laplacian and commute-time kernels computed from the cosine similarity. Parameter k is chosen by cross validation using training data.

We also employ Naive-Bayes classifier (NB) for multi-class classification, as another baseline.

It is worth noting the large number of categories in the datasets (52 and 30). This makes it difficult to apply support vector machines and other high-performance classifiers for binary classification.

Multi-label classification In multi-label classification tasks, a document may be associated with one or more categories. For these tasks, we use the Enron and the Bibtex datasets[6]. The classification procedure follows that of multi-class classification, with one exception that we use the ML-$k$nn algorithm (Zhang and Zhou 2007) in place of $k$nn classification.

[3]We limited the abstracts to those published in year 2000 or later.

[4]http://csmining.org/index.php/r52-and-r8-of-reuters-21578.html

[5]http://www.zjucadcg.cn/dengcai/Data/TextData.html

[6]http://mulan.sourceforge.net/index.html

Table 1: Experimental Results. The rows marked MeSH A–D show the results (average highest ranking of family terms; see text for definition) of ranking tasks, Reuters-52 and TDT2-30 show the average error rates in multi-class classification tasks and Enron and Bibtex correspond to the average label disagreement in multi-label classification tasks. In all tasks, smaller the values, the better. Cos, $\mathbf{L}_{\mathrm{RL}}$, $\mathbf{L}_{\mathrm{CT}}$, and NB respectively stand for cosine, regularized Laplacian, the commute-time kernels, and Naive Bayes classifiers. $\mathbf{L}_{\mathrm{CT}}$ dist is the commute-time distance obtained with the application of Eq. (9) to $\mathbf{L}_{\mathrm{CT}}$. The figures in parenthesis under $\mathbf{L}_{\mathrm{RL}}$ show the value of $\beta\lambda_n$, where λ_n is the spectral radius of Laplacian $\mathbf{L}$.

Dataset		Cos	$\mathbf{L}_{\mathrm{RL}}$							$\mathbf{L}_{\mathrm{CT}}$	$\mathbf{L}_{\mathrm{CT}}$ dist	NB	# objects
			($\beta\lambda_n$= 0.01)	(0.1)	(0.5)	(1)	(10)	(100)	(1000)				(# features)
MeSH A	Skewness	6.6203	6.5802	6.1549	4.4874	3.3225	1.1931	0.9554	0.9294	**0.9188**	8.9454	-	833
	Rank	14.7	14.7	14.5	14.2	13.9	**13.4**	13.6	13.7	13.7	172.9	-	(274064)
MeSH B	Skewness	9.9111	9.8021	8.6242	5.7524	3.835	2.1594	1.7736	1.7664	**1.7457**	14.37	-	2098
	Rank	42.6	42.6	42.4	42.0	41.6	39.4	38.6	**38.5**	**38.5**	382.1	-	(228522)
MeSH C	Skewness	7.3111	7.2594	6.5986	4.6353	3.3799	**0.9770**	1.0942	1.2097	1.2154	11.466	-	1347
	Rank	42.0	42.0	41.8	41.2	40.6	38.7	37.7	**37.4**	**37.4**	284.2	-	(200339)
MeSH D	Skewness	9.0052	8.9104	8.3939	6.3507	4.8183	**1.4867**	1.5200	1.5781	1.5753	13.886	-	1961
	Rank	119.0	118.9	118.7	117.5	116.4	110.4	106.6	105.9	**105.7**	438.1	-	(212614)
Reuters-52	Skewness	14.815	14.721	14.318	12.722	11.044	**6.1597**	6.7267	6.9076	6.9341	30.1115	-	9100
	Error rate	0.153	0.153	0.148	0.131	0.128	0.107	0.102	0.107	**0.100**	0.578	0.135	(19241)
TDT2-30	Skewness	3.6291	3.6145	3.4309	2.8966	**2.5600**	3.3985	4.1023	4.2001	4.2027	30.5958	-	9394
	Error rate	**0.036**	**0.036**	**0.036**	0.037	**0.036**	0.039	0.042	0.042	0.037	0.531	0.039	(36771)
Enron	Skewness	6.4651	6.3987	5.7334	3.6943	2.7401	**2.5742**	2.9286	3.0307	3.0375	12.8882	-	1694
	Disagreement	2.80	2.79	2.70	2.65	2.69	**2.61**	2.68	2.64	2.67	3.30	-	(1001)
Bibtex	Skewness	2.4726	2.4620	**2.4306**	2.6106	2.9568	4.7303	5.6225	5.7226	5.7345	27.1330	-	7395
	Disagreement	**1.93**	**1.93**	**1.93**	1.94	1.95	1.95	1.97	1.97	1.97	2.37	-	(1836)

The number of unique assignment of category combination to an object is 753 in Enron, and 2856 in Bibtex, which are again extremely large.

Evaluation metrics For all tasks, we compare cosine similarity (Cos) with the regularized Laplacian ($\mathbf{L}_{\mathrm{RL}}$) and commute-time kernels ($\mathbf{L}_{\mathrm{CT}}$), in terms of the degree of hub emergence and the task performance.

Following Radovanović et al. (Radovanović, Nanopoulos, and Ivanović 2010a), we evaluate the degree of hubness by the skewness of the N_{10} distribution, which is defined as

$$S_{N_{10}} = \frac{\mathbb{E}[N_{10} - \mu_{N_{10}}]^3}{\sigma_{N_{10}}^3},$$

where $\mathbb{E}[\cdot]$ is the expectation operator, and $\mu_{N_{10}}$ and $\sigma_{N_{10}}^2$ are the mean and the variance of the N_{10} distribution, respectively. Larger skewness indicates a stronger emergence of hubs in the data.

In the ranking tasks with MeSH thesaurus, for each term treated as a query, we rank all other terms in the thesaurus, sorting them by their similarity to the query. Then we evaluate the performance of the similarity measure, by the highest rank of terms that are the "family members" of the query term. Here, a term is a family member of a query term if it is the parent, a child or a sibling of the query term in the MeSH tree. Because these are the terms that can be regarded as semantically closest to the query, a sensible similarity measure should rank them higher than other terms in the ranking list for the query term; in an ideal case, the term ranked no. 1 should be a family member. Hence, in this task, smaller this performance metric (i.e., the highest rank of a family member term), the better. The results are averaged over all terms (queries) in the MeSH tree.

In multi-class classification, we calculate the error rate of predicting the correct category of test documents (smaller the better). In multi-label classification, we count the number of disagreement between the correct categories and the predicted ones for each test document (smaller the better). The results are then micro-averaged over all test documents.

Results Experimental results are shown in Table 1. Although we omit the proof due to space restriction, it can be shown that the off-diagonal elements of the regularized Laplacian ($\mathbf{L}_{\mathrm{RL}}$) matrix become proportional to those of cosine similarity (Cos) matrix as β approaches to 0, and to those of the commute-time kernels ($\mathbf{L}_{\mathrm{CT}}$) as β tends to infinity. For this reason, we place the results for $\mathbf{L}_{\mathrm{RL}}$ between those of Cos and $\mathbf{L}_{\mathrm{CT}}$ in Table 1.

For all MeSH categories A–D in the ranking task, $\mathbf{L}_{\mathrm{RL}}$ and $\mathbf{L}_{\mathrm{CT}}$ show lower skewness compared to Cos, and simultaneously improve the performance (the 'Rank' rows showing the averaged highest rank of family terms). Note that a smaller rank shows a better performance.

In multi-class classification tasks, and in particular on Reuters-52 dataset, cosine similarity (Cos) shows high skewness. With $\mathbf{L}_{\mathrm{RL}}$, skewness in Reuters-52 first decreases as the parameter β is increased, but skewness then starts to increase modestly at some point. Performance (error rate) is also improved with the increase of β, and even after skewness turns to increase, error rate continues to improve. For this dataset, $\mathbf{L}_{\mathrm{CT}}$ achieves the least error rate. It also outperforms the naive Bayes (NB) classifier. On TDT2-30 dataset, skewness again drops but then goes up as parameter β is increased, this time to a value higher than that of cosine similarity. The error rate remains nearly constant, which tells us that Laplacian-based kernels are not effective for this

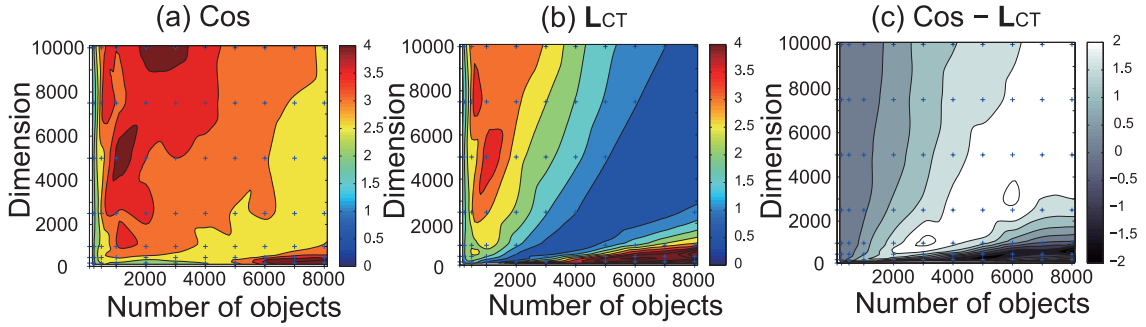


Figure 3: Contour plots of the skewness of the N_{10} distributions, computed for synthetic datasets generated with varying number of objects and feature dimensions (+ mark corresponds to a dataset). Panels (a) and (b) show skewness of cosine similarity and the commute-time kernels, respectively, and panel (c) shows the difference of skewness between them.

dataset.

In multi-label classification tasks with the Enron dataset, the regularized Laplacian $\mathbf{L}_{RL}$ and the commute-time kernels $\mathbf{L}_{CT}$ show skewness lower than cosine similarity. As skewness decreases, the performance (disagreement) also improves. For the Bibtex dataset, however, skewness of the Laplacian-based kernels is mostly higher than that of cosine similarity. The performance (disagreement) remains more or less identical to that of Cos, only slightly worse.

In Table 1, we also show the results of using commute-time distance in column '$\mathbf{L}_{CT}$ dist', which is the distance in the feature space of the commute-time kernels $\mathbf{L}_{CT}$, computed by Eq. (9). For all datasets, the extremely high skewness and poor performance of '$\mathbf{L}_{CT}$ dist' suggest that commute-time *distance* in fact promotes hubs, and its use with *k*nn classification is not a good idea.

Discussion

Skewness for parameter tuning

The experimental results in the previous section shows that, contrary to our expectation, Laplacian-based kernels do not always reduce hubs. Skewness, which is an indicator of hubness, decreased in the MeSH datasets in ranking tasks, Reuters-52 dataset in multi-class classification and Enron dataset in multi-label classification. It did not decrease in Bibtex and TDT2-30 datasets.

However, in the tasks in which Laplacian-based kernels indeed decreased skewness, improved task performance was obtained. Moreover, the kernel giving the smallest skewness value attains the best task performance, or is very close to the best, among the tested kernels and parameters. This result suggests a way to choose suitable kernels, and to automatically tune parameters of the regularized Laplacian, by using skewness as an indicator of kernel performance.

Hubness phenomenon and dataset size

Let us now discuss hubness from the viewpoint of dataset size (the number of objects), in a simulation with synthetic datasets.

We create synthetic datasets of sparse vectors in the same way as we generated Figures 1 and 2. In these figures, the number of objects n was 500, and the dimension d was 10 or 50. Here, we vary n between 100 through 8000, and d between 100 to 10000. The $n \times n$ of cosine similarity matrix is then calculated with the n objects of d dimensional vectors. Finally, we compute (as the representative of the Laplacian-based kernels) the commute-time kernels matrix, just as we did previously.

Using cosine similarity and the commute-time kernels as similarity measures, we obtain skewness of N_{10} distribution. We use averaged skewness over 10 times repetition of each combination of n and d.

Figure 3 shows contour plots of skewness: (a) cosine similarity (Cos), (b) commute-time kernels ($\mathbf{L}_{CT}$) and (c) difference in skewness between cosine similarity and commute-time kernels. (Cos $-$ $\mathbf{L}_{CT}$). The vertical axis shows the number of objects n and the horizontal axis shows the number of features (dimension) d.

From the figure, we observe the following. First, for cosine similarity shown in panel (a), emergence of hubs depends not only on feature dimension (as Radovanović et al. reported) but also on the number of objects. And comparing panel (a) and (b), we see that the commute-time kernels (panel (b)) shows smaller value of skewness in a larger area (for various number of object and feature dimensions) than that of cosine similarity (panel (a)). Second, panel (c) shows that by converting cosine similarity matrix into the commute-time kernels, skewness is reduced for various dimensions of objects and features. However, when datasets consist of large number of objects and small number of features, such as $n > 5000, d < 1000$, skewness increases and hubs emerge more than cosine similarity. We presume that this may be related to the increase of skewness with Bibtex dataset, in which the number of objects is 7395 and features is 1836.

Commute-time distance

Laplacian-based kernels are sometimes used to compute distance, through the translation of Eq. (9). In particular, the distance computed from the commute-time kernels (known as *commute-time distance* or *resistance distance*) has a nice interpretation that it is proportional to the expected number of steps a random walk has to take to go from one vertex to

another vertex for the first time and then coming back. However, Brand (Brand 2005) and Fouss et al. (Fouss et al. 2006) independently reported that the commute-time distance was less effective than the inner products given by the commute-time kernels on collaborative filtering tasks.

Our experimental results agree with their reports; using commute-time distance deteriorated the performance in all the experiments we conducted.

Regarding commute-time distance, von Luxburg et al. (von Luxburg, Radl, and Hein 2010) show that as the number of objects in the database increases, k-nearest neighbor lists become nearly identical for all objects. Thus, regardless of the query, the same objects appear in the k-nearest neighbor lists. This phenomenon is observed throughout all datasets used in our experiment, and resulted in a strong correlation between the number of objects in the dataset and the skewness (see Table 1).

Conclusion

In this paper, we have pointed out that Laplacian-based kernels make all objects in the dataset equally similar to the centroid. With this observation, we have investigated whether Laplacian-based kernels, in particular commute-time kernels and regularized Laplacian, reduce hubs in high-dimensional data. They worked quite well in ranking tasks, but in classification tasks (multi-class and multi-label classification), the results were mixed; in some tasks and datasets, they slightly reduced skewness but did not lead to performance improvement.

However, whenever these kernels indeed reduced skewness, the kernel that achieves the smallest skewness performed best or close to the best among all the tested kernels. This result suggests that skewness could be used as a yardstick of kernel performance. Note that because skewness can be computed without any label information, its evaluation can be done in an unsupervised manner.

We also found that when we use Laplacian-based kernels, it is almost always better to use the Gram matrix as it is as a similarity matrix, than to translate them into distance in the feature space, both in terms of skewness of N_{10} distribution and performance.

Besides using Laplacian-based kernels, we can also make the objects equally similar to the centroid by centering the original similarity matrix. Hence we expect centering to also reduce hubs, and plan to investigate this subsequently.

Acknowledgments

We thank anonymous reviewers for valuable comments.

References

Beyer, K. S.; Goldstein, J.; Ramakrishnan, R.; and Shaft, U. 1999. When is "nearest neighbor" meaningful? In *Proceedings of the 7th International Conference on Database Theory (ICDT '99)*, 217–235. Springer.

Bishop, C. M. 2006. *Pattern Recognition and Machine Learning*. Springer.

Brand, M. 2005. A random walks perspective on maximizing satisfaction and profit. In *Proceedings of the 5th SIAM International Conference on Data Mining (SDM '05)*, 12–19.

Chebotarev, P. Y., and Shamis, E. V. 1997. The matrix-forest theorem and measuring relations in small social groups. *Automation and Remote Control* 58(9):1505–1514.

Chung, F. R. K. 1997. *Spectral Graph Theory*. American Mathematical Society.

Fouss, F.; Yen, L.; Pirotte, A.; and Saerens, M. 2006. An experimental investigation of graph kernels on a collaborative recommendation task. In *Proceedings of the 6th International Conference on Data Mining (ICDM '06)*, 863–868.

François, D.; Wertz, V.; and Verleysen, M. 2007. The concentration of fractional distances. *IEEE Transactions on Knowledge and Data Engineering* 19:873–886.

Hastie, T.; Tibshirani, R.; and Friedman, J. 2001. *The Elements of Statistical Learning*. Springer.

Kondor, R. I., and Lafferty, J. 2002. Diffusion kernels on graphs and other discrete input spaces. In *Proceedings of the 19th International Conference on Machine Learning (ICML '02)*.

Radovanović, M.; Nanopoulos, A.; and Ivanović, M. 2010a. Hubs in space: popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research* 11:2487–2531.

Radovanović, M.; Nanopoulos, A.; and Ivanović, M. 2010b. On the existence of obstinate results in vector space models. In *Proceedings of the 33rd ACM SIGIR Conference (SIGIR '10)*, 186–193.

Saerens, M.; Fouss, F.; Yen, L.; and Dupont, P. 2004. The principal components analysis of graph, and its relationships to spectral clustering. In *Proceedings of the 15th European Conference on Machine Learning (ECML '04)*, 371–383.

Smola, A. J., and Kondor, R. 2003. Kernels and regularization on graphs. In *Proceedings of the 16th Annual Conference on Learning Theory and 7th Kernel Workshop*, 144–158.

von Luxburg, U.; Radl, A.; and Hein, M. 2010. Getting lost in space: large sample analysis of the resistance distance. In *Advances in Neural Information Processing Systems 23*, 2622–2630.

Zhang, M.-L., and Zhou, Z.-H. 2007. ML-KNN: a lazy learning approach to multi-label learning. *Pattern Recognition* 40:2038–2048.

Zhou, D.; Bousquet, O.; Lal, T. N.; Weston, J.; and Schölkopf, B. 2004. Learning with local and global consistency. In *Advances in Neural Information Processing Systems 16*, volume 1, 595–602. MIT Press.

Zhu, X.; Ghahramani, Z.; and Lafferty, J. 2003. Semi-supervised learning using Gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine Learning (ICML '03)*.