Affinity Learning via Self-diffusion for Image Segmentation and Clustering

Bo Wang[1] and Zhuowen Tu[2,3]
[1]Department of Computer Science, University of Toronto
[2]Microsoft Research Asia
[3]Lab of Neuro Imaging and Department of Computer Science, UCLA
{wangbo.yunze, zhuowen.tu}@gmail.com,

Abstract. Computing a faithful affinity map is essential to the clustering and segmentation tasks. In this paper, we propose a graph-based affinity (metric) learning method and show its application to image clustering and segmentation. Our method, self-diffusion (SD), performs a diffusion process by propagating the similarity mass along the intrinsic manifold of data points. Theoretical analysis is given to the SD algorithm and we provide a way of deriving the critical time stamp t. Our method therefore has nearly no parameter tuning and leads to significantly improved affinity maps, which help to greatly enhance the quality of clustering. In addition, we show that much improved image segmentation results can be obtained by combining SD with e.g. the normalized cuts algorithm. The proposed method can be used to deliver robust affinity maps for a range of problems.

1. Introduction

Graph-based approaches for segmentation [26] and manifold learning [24, 30] build their success on top of meaningful pair-wise distances/metrics which can equivalently be represented as affinity maps/similarity measures; they are further generalized into Laplacian graphs [2, 38]. Therefore, constructing a faithful metric becomes crucial in a variety of Laplacian graph-based applications [26, 10, 37].

A large body of literature on manifold learning [30, 24, 5] explicitly construct new embedding spaces with an enhanced distance notion. A recent line of work in machine learning introduces the idea of distance propagation/diffusion [37, 5], allowing the derived distances to follow the intrinsic data manifolds; moreover, they do no need to explicitly construct the geometry of the manifolds, which is typically a computation demanding task. This situation is more tempting when the output by a particular algorithm [3] only includes the pairwise distance measures with no explicit output features. Other examples include Markov random walks on manifolds [29] and Ranking on Data Manifolds (RDM) [36]. In particular, RDM uses a similarity-induced diffusion kernel to improve the ranking result with respect to a single query. Although the above methods have been shown to be effective in the classification/ranking tasks, they lack the notion of a global similarity metric, which is crucial in many applications.

Along the line of diffusion-based metric learning, diffusion maps (DM) [5] defines diffusion distances between data samples. An input similarity matrix is then improved through a diffusion process. There is an explicit notion of a global distance metric in DM and the diffusion step t provides a way of doing multi-scale data analysis. More recently, a self-smoothing operator (SSO) is introduced in [16] which is a diffusion-based unsupervised metric learning approach and related to diffusion maps [5]. However, instead of using the notion of diffusion distances between data samples, SSO works on the similarity matrix/affinity map directly using a self-induced smoothing kernel. Both DM and SSO have the tuning parameter t to specify; in general, it is critical to have a principled way of deciding t, which is lacking in the current DM and SSO.

Once a faithful affinity map/similarity matrix is learned, various graph Laplacian-based applications can be performed. Here, we focus on the task of clustering (unsupervised), image segmentation (unsupervised), and interactive image segmentation (semi-supervised), which are all dependent on accurate affinity maps. Image segmentation has been a long-standing problem and typical approaches include Normalized Cuts [26], DDMCMC [32], Mean-shift [6], and fast graph-based approach [11]. Dynamic-based methods(e.g.DM, SSO) are difficult to be applied in segmentation due to two reasons: (1) Dynamic-based methods require a large computation cost which hinder them dealing with large scale segmentation; (2) Derivation of a proper iteration number is lacked by most dynamic-based methods. Fig.(1) illustrates the effect of iteration steps, and our method provides a closed-form reasonable approximation of the best iteration steps.

In this paper, we focus on developing an effective algorithm to deliver enhanced affinity maps for clustering and image segmentation. Our method is related to distance propagation/diffusion methods such as DM [5] and SSO

[16]. Instead of computing the distances on the diffused probability transition kernel [5], we define our diffusion kernel in the original space [16] but with a new way of performing the diffusion. More importantly, we provide theoretical analysis to our approach and show how to derive time step t, which are lacking in both DM and SSO; in addition, we develop a fast self-diffusion process and show how SD can be directly combined with normalized cuts [26] for the image segmentation task, while DM and SSO have a practical issue of performing segmentation on a graph of medium or high resolution due to their prohibitive computation cost.

2. Related Work

Suppose we are given a graph $G = (\Omega, W)$ where $\Omega = \{x_i\}_{i=1}^n$ is a collection of the data samples; W is a similarity matrix with each element $W(i,j) \in [0,1]$ being the similarity between sample x_i and x_j. $W(i,j) = 1$ is interpreted as having the identical x_i and x_j; $W(i,j) = 0$ suggests that x_i and x_j are being completely dislike. W is often obtained by applying the Gaussian kernel to a distance matrix:

$$W(i,j) = \exp\left\{-d^2(i,j)/(\sigma^2)\right\} \tag{1}$$

where $d(i,j)$ denotes the distance between x_i and x_j, and σ decides the kernel size. A stochastic diffusion process on G allows local similarities to be propagated along the data manifold without the need to explicitly construct the manifold geometry. Diffusion/propagation based approaches can be roughly categorized into two types: equilibrium-based and dynamics-based. In this paper, we focus on dynamics-based approaches.

The transition kernel is given by row-wise-normalizing the similarity matrix W which encodes the outbound link information:

$$P = D^{-1}W \tag{2}$$

where D is a diagonal matrix with $D(i,i) = \sum_{k=1}^n W(i,k)$.

One example of dynamics-based approaches is label propagation [37], which is widely used to solve the semi-supervised learning problem. It propagates each labeled sample's label information to its neighboring samples. Here we give a formulation in the context of retrieval [1]. Label propagation uses the same row-wise normalized transition kernel (2). With an initial $f_0 = y$, it iterates between the following two steps: (1). $f_{t+1} = Pf_t$; (2). $f_{t+1}(i) = 1$ if $y_i = 1$. Actually, because P is row-wise normalized, we have $\mathbf{1} = P\mathbf{1}$ where $\mathbf{1}$ denotes a constant vector. Therefore the iterations have to be stopped before the equilibrium is achieved, and the step parameter t provides a natural way of doing multi-scale data analysis.

Diffusion map (DM) [5] is another dynamics-based approach and it introduces a global distance metric (i.e. diffusion distances) over data samples. Given the transition kernel P in (2), the diffusion distance between x_i and x_j at step t is defined as:

$$\begin{aligned} d_t^2(i,j) &= ||p_t(i,\cdot) - p_t(j,\cdot)||^2_{1/\phi_0} \\ &= \sum_{k=1}^n \frac{1}{\phi_0(k)}(p_t(i,k) - p_t(j,k))^2 \end{aligned} \tag{3}$$

where $p_t(i,\cdot)$ is the i-th row of the t-step transition matrix $P_t = P^t$, and ϕ_0 is the equilibrium distribution. It can be shown that the diffusion distance can be directly computed from the diffusion map: $\Psi_t : x_i \rightarrow \left[\lambda_1^t\psi_1(i), \cdots, \lambda_q^t\psi_q(i)\right]^T$, and

$$d_t^2(i,j) \approx \|\Psi_t(i) - \Psi_t(j)\|^2 = \sum_{k=1}^q \lambda_k^{2t}(\psi_k(i) - \psi_k(j))^2.$$

Here ψ_k's are the right eigen-vectors of P, and $q \leq n-1$ captures the leading nontrivial eigenvalues. Since $p_{t\rightarrow\infty}(i,\cdot) = \phi_0^T$, at the equilibrium point any pairwise diffusion distance is 0, and what matters is the dynamics of the diffusion process. The definition of the diffusion distances in DM however still lacks of thorough theoretical justification.

Like in the diffusion map method [5], self-smoothing operator (SSO) [16] also performs the dynamic propagation; however, SSO computes the similarity after running random walk for t times on the original affinity map:

$$P = D^{-1}W \;\; and \;\; W_t = WP^t.$$

Therefore, SSO reduces the redundant steps in DM to covert $W \rightarrow P$, $P \rightarrow d$, and $d \rightarrow W$, which causes degraded accuracy in the affinity map and introduces extra uncertainties in the parameter tuning. Large improvement of SSO over DM is observed in [16] and this justifies the benefits of working on W directly. However the results in the SSO [16] are mostly empirical and, like in the DM, the critical parameter t is left untouched. The differences between SSO and self-diffusion (SD) are three-fold: 1) SSO converges to a uniform matrix while SD converges to a pseudo-inverse of the graph Laplacian; 2) the most critical parameter t in DM and SSO was empirically set, 3)SSO cannot deal with large matrix, and therefore no thorough real-size segmentation was presented in SSO, while SD can be efficiently performed by eigen-approximation(see Section(4.3)).

3. Self-Diffusion

Here we introduce a new diffusion operator, self-diffusion process (SD). It works directly on the input affinity map and uses a regularization in each diffusion iteration.

Fig. 2 shows the basic algorithm of the self-diffusion (SD) algorithm. As in other manifold learning algorithms, SD is based on the assumption that long-range similarities

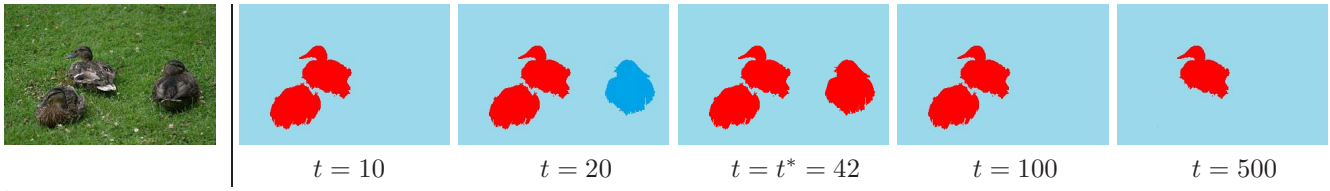


Figure 1: A qualitative comparison of segmentation results via self-diffusion w.r.t. different number of iterations. t^* is derived in Theorem 4.4. After 500 times, the result remains almost constant.

can be approximated by an accumulation of local similarities. SD is not expected to work well when the assumption fails or the local similarities can not be reliably estimated.

The computed affinity/similarity map W^* can be used in numerous applications. For example, given W^* retrieval can be done on a per-row basis [1]. In other words, for each row (a query), the similarity scores are sorted in descending order and the first K (retrieval window) items are returned. As stated previously, graph-based Laplacian approaches [26, 10, 37] including clustering, segmentation, and semi-supervised learning can all benefit from having an accurate affinity map.

1. Computing the smoothing kernel:

$$P = D^{-1}W$$

where D is a diagonal matrix with $D(i,i) = \sum_{k=1}^{n} W(i,k)$.

2. Performing smoothing for t^* steps:

$$W_t = W_{t-1}P + I.$$

3. Self-normalization: $W^* = W_t * D^{-1}$

Figure 2: Algorithm of self-diffusion (SD).

Given an input similarity matrix W, the only parameter in SD is the step t. As in other dynamics-based approaches, t controls the scale at which the data are analyzed. We have an analogy to image de-noising, performing diffusion on a noisy image naturally increases the signal-to-noise ratio. However, if too much diffusion is done, the image becomes over-smoothed and too much information is lost. In this regard, t in SD has the same effect in improving the similarity (See Fig.(1)). We provide a close form for estimating an appropriate t^* at which the effect of SD is the best (see Lemma 4.1). Eventually SD will converge to a degenerated (over-smoothed) point where the segmentation result stays constant over all queries. However, we prove that our algorithm converges to the general inverse of the Laplacian matrix of the input similarity, which is still a valid kernel.

4. Theoretical Analysis

One may wonder if there is a closed form for self-diffusion process. The following lemma illustrates the closed form for self-diffusion process:

Lemma 4.1. *The closed form of self-diffusion transition matrix at step* t *is as follows:*

$$W_t = (I - P^t)(I - P)^{-1} + WP^t \tag{4}$$

Proof. We assume another iterative process $S_t = S_{t-1} * P + I$ with $S_0 = I$. We can verify that

$$S_t = I + P + \cdots + P^t = (I - P^{t+1})(I - P)^{-1}. \tag{5}$$

Let $Z_t = W_t - S_t$ with $Z_0 = W - I$. Hence we have

$$Z_t = (W_t - S_t) = (W_{t-1} - S_{t-1})P = Z_{t-1}P \tag{6}$$

It is easy to derive that $Z_t = Z_0P^t = (W - I)P^t$, which leads us to

$$\begin{aligned} W_t &= S_t + Z_t \\ &= I + P + \cdots + P^t + (W - I)P^t \\ &= I + P + \cdots + P^{t-1} + WP^t \\ &= (I - P^t)(I - P)^{-1} + WP^t \end{aligned} \tag{7}$$

□

The following lemma shows the property of convergence of self-diffusion process $\lim_{t\to\infty} W_t$:

Lemma 4.2. $\lim_{t\to\infty} W_t = (I - P)^{-1}$. *And after self-normalization, i.e.,* $\lim_{t\to\infty} W^* = W_t * D^{-1}$, W^* *is a valid kernel.*

Proof. Note that $I - P$ has one zero eigenvalue, so $(I - P)^{-1}$ is actually a Moore-Penrose pseudo-inverse of $I - P$. To prove $\lim_{t\to\infty} W_t = (I - P)^{-1}$, we need to prove that $\lim_{t\to\infty}(I - P)W_t(I - P) = (I - P)$. Use Lemma(4.1), we have

$$(I - P)W_t(I - P) = I - P + (W - PW - I)(P^t - P^{t+1})$$

According to the convergence of Markov Chain, we have $lim_{t\to\infty}(P^t - P^{t+1}) = \mathbf{0}$, hence, we can get $\lim_{t\to\infty}(I - P)W_t(I - P) = (I - P)$, then $\lim_{t\to\infty} W_t = (I - P)^{-1}$.

After self-normalization, we have

$$\begin{aligned} W^* &= (I - P)^{-1}D^{-1} = (D - DP)^{-1} \\ &= (D - W)^{-1} = L^{\dagger} \end{aligned} \tag{8}$$

where $L = D - W$ is the graph Laplacian matrix, and $L^\dagger$ means the Moore-Penrose pseudo-inverse of the graph Laplacian, also called inverse Laplacian. Inverse Laplacian has been greatly explored recently [12]. It is symmetric, semi-definite and also a gram matrix. These properties make the inverse Laplacian a valid kernel. We only provide a brief proof of these properties. Since $L = D - W$ is symmetric, and by definition, $L^\dagger = (L^T L)^{-1} L$, so $L^\dagger$ is also symmetric. We know L is positive semi-definite.Let the eigen-decomposition of L be QVQ^T, then $L^\dagger = QV^{-1}Q^T$, which shows that the eigenvalues of L and $L^\dagger$ have the same sign. Hence $L^\dagger$ is positive semi-definite. Since $L = QVQ^T = (QV^{1/2})(QV^{1/2})^T = UU^T$, where $U = QV^{1/2}$, we see that $L^\dagger = (U^\dagger)(U^\dagger)^T$. □

4.1. The relation to random walks

One related model with the proposed method is Markov random walks [29]. Assume the transition matrix is A,we can simply use a matrix power to calculate state probability of the random walks at step t:

$$P_{t|0}(k|i) = [A^t]_{ik}$$

This transition matrix is helpful in semi-supervised learning. With the Bayes' Rules, we can have the posterior probability of the label for point k is given by

$$P_{post}(y|k) = \sum_i P(y|i) P_{t|0}(i|k) \tag{9}$$

The difference between our method and random walks is the use of P and W at the same time. The initial kernel information W is helpful to improve the performance during the iteration. In essence, due to Lemma 4.1, we have :

$$\frac{\partial W_t}{\partial W} = P^t \tag{10}$$

which means that our self-diffusion contains a random walks on W. When t is large, random walks loses all the information of the initial transition probability A, and have the property of slow information loss $P^{2t} \approx P^t$. As to the self-diffusion, we have the following property:

Lemma 4.3. *The information loss rate of self-diffusion process is proportional to the one in random walks.*

Proof. We compare the difference of $W_{2t} - W_t$ with $P^{2t} - P^t$. Using Eqn.(4), we have

$$W_{2t} - W_t = [W - (I - P)^{-1}](P^{2t} - P^t) \tag{11}$$

which means the information loss of self-diffusion is proportional with random walks. This can be also seen by the fact that:

$$\frac{\partial W_t}{\partial P^t} = W - (I - P)^{-1} \tag{12}$$

We call the matrix $W - (I - P)^{-1}$ as the "random walks coefficient matrix". □

The choice of iterative number t is essential. With the help of labeled data, cross validation can be used to determine t, which needs a lot of labeled data. For unsupervised learning, we give a heuristic method to determine the optimal t in the following section.

4.2. The Choice of t^*

As we discussed before, when t is too large, the similarity function becomes too smooth to be discriminative. Usually, the optimal iteration number of t is manually decided. Here we provide an automatic way to decide the number of t. The literature of determining a right number of iteration t in a iterative algorithm is rare. [29] uses the labeled data to choose the number t which maximize the margin between all the labeled classes. Dissipation of mutual information [31] is adopted to choose t. Assume the mutual information is given by

$$I(t) = I(W_t; W) = \sum_j p_j \sum_i P_{ij}^t \log \frac{P_{ij}^t}{p_i^t}$$

where p_j is the prior probability of the states, and $p_i^t = \sum_j p_{ij}^t p_j$ is the unconditioned probability of data x_i at time t. The criteria used in [31] is

$$t^* = sup\{-\frac{I(t)}{dt} > 0\}$$

However, this choice of t suffers from the fact that the computation of $-\frac{I(t)}{dt}$ has no closed form and has to be calculated in every iteration.

In this paper, we propose a new heuristic to choose optimal t for self-diffusion. For a degenrated similarity matrix W with the diagonal elements not being exactly 1 (in our case due to the self-diffusion proces), we define the "degree of freedom"(DoF) as :

$$DoF(W) = Tr(W) \tag{13}$$

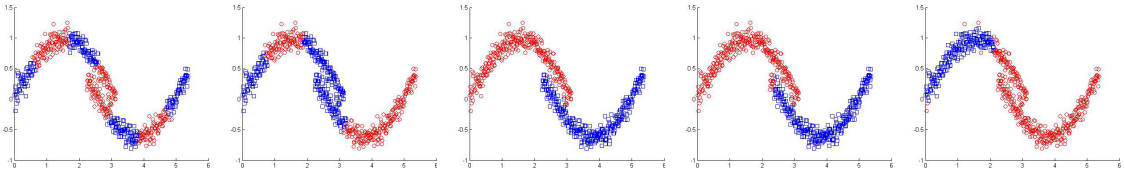
We use the concept in functional analysis that $\nabla DoF(W)$ is a indicator of smoothness of $DoF(W)$, where ∇ is a Laplace operator. We run our self-diffusion on a toy data (shown in Fig.3(a)) and report the accuracy of the clustering and the value of $\nabla DoF(W_t)$ as iteration numbers in Fig.3(b). This is a common observation that the similarity has the most discriminative power when the degree of freedom of diffusion process has started to own a smooth evolvement, i.e., $\nabla DoF(W_t)$ starts to be null.

Hence, we let the stopping rule to be

$$t^* = \min\{t : \nabla DoF(W_t) \leq \varepsilon\} \tag{14}$$

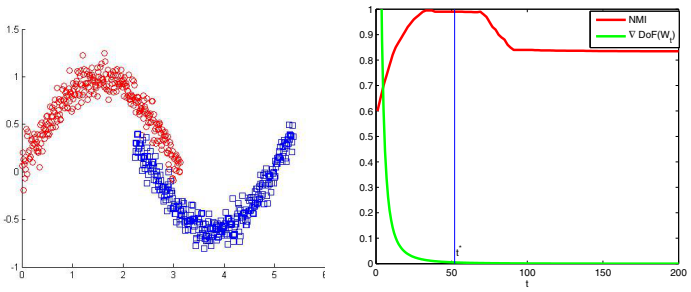
where ε is a small positive constant.

The following theorem gives an estimate of the optimal t^*:



$t = 10$ $t = 30$ $t = t^* = 52$ $t = 80$ $t = 150$

Figure 4: Clustering results w.r.t. to the number of iterations.



(a) (b)

Figure 3: (a) Toy data. (b) The accuracy of the clustering results (measured by Normalized Mutual Information(NMI) [28]) over a long range of iterations.

Theorem 4.4. *If we denote $\lambda_1 > \lambda_2 \ldots, > \lambda_n$ the eigenvalues of the transition matrix P, a good estimate of t^* is*

$$t^* = \lceil \ln(\frac{\epsilon}{(n - DoF(P))\|(W - WP - I\|)} / \ln(\lambda_2 - C) \rceil. \quad (15)$$

We give a brief derivation of this theorem in the appendix. In this paper, a typical value $C = 0.05$ is used. Fig.(4) shows the clustering results w.r.t. different number of iterations. We can see that our approximation of stopping time t^* is a good approximation.

4.3. Fast Self-Diffusion

The algorithm of self-diffusion in Fig. 2 is an iterative dynamic process which poses a certain amount of computational burdens. For a small or medium scale dataset, the computation cost is acceptable, but for large-scale dataset, its computational requirement might be prohibitive. Instead, the theoretical analysis of this algorithm in Lemma 4.1 provides us an analytic form to calculate the final similarity matrix, like in DM. However, the needed calculation of $(I - P)^{-1}$ and P^t is also a burden. Here we propose a fast method to calculate the affinity map induced by the algorithm of self-diffusion.

For the transition matrix $P \in R^{n \times n}$, we calculate the eigen-decomposition V, U such that $PV = VU$, where $U \in R^{K \times K}$ is the diagonal matrix whose entries are the biggest K eigenvalues of P, and $V \in R^{n \times K}$ is the corresponding set of orthogonal eigenvectors. We achieve this standard eigen-decomposition using the eigen-solver in [26]. It is shown that the cost is less than $\mathcal{O}(n^2)$. It is easy to verify that $P^t V = VU^t$, and $(I - P^t)(I - P)^{-1}V = V(I - U^t)(I - U)^{-1}$. Hence, if V is normal, then we can estimate the final matrix as follows:

$$W_t \approx V(I - U^t)(I - U)^{-1}V' + DVU^{t+1}V' \quad (16)$$

The calculation of U^t costs $\mathcal{O}(tK)$, the calculation $V(I - U^t)(I - U)^{-1}V'$ needs $\mathcal{O}(Kn^2)$, so is $DVU^{t+1}V'$. If $K \ll n$, the overall time complexity is $\mathcal{O}(n^2)$. For clustering and segmentation tasks in the experiments, we set $K = 20$.

5. Experiments

In this section, we show the performance of self-diffusion (SD) in learning an effective affinity map on applications of clustering (unsupervised), image segmentation (unsupervised), and interactive segmentation (semi-supervised).

Since a main contribution of this paper is the derivation of t^*. We briefly summerize its importance first. Note that in [16], for the SSO on the SC+IDSC distance on MPEG7, the t^* is manually picked as $1,000$ with the corresponding accuracy 97.64%, while in the ANDI dataset, the best t is 0.75 with accuracy 91.67%. Using $t = 0.75$ and $1,000$ for MPEG7 and ANDI will otherwise achieve unsatisfactory 93% and 40% respectively. This shows the large difference and importance in t^*. To compare, SD automatically calculates the proper t* to be 52 with accuracy of 98.24%; also, on the ANDI dataset, $t^* = 22$ with accuracy 92.50% (t values in SD and SSO not directly comparable due to the algorithm difference). This immediately shows the importance of our contributions here.

5.1. Clustering

We apply SD to face and 3D image clustering. In both cases, raw pixels are used directly. The similarity between two images i and j is calculated as Eqn.(1) and we set $\sigma = 0.3$ for all clustering experiments.

5.1.1 Face Clustering

In this section, we test our method on the face clustering task. Yale face dataset B [14] is a standard benchmark dataset for face clustering, which consists of $5,760$

Table 1: Quantitative comparison of the proposed methods with other standard clustering methods on the Yale B dataset [14]. ↑ prefers higher values whereas ↓ appreciates lower scores.

Methods/Score	F1 ↑	NMI ↑	CPU(s) ↓
Kernel K-means	0.67	0.73	2.3
Spectral Clustering	0.69	0.79	3.2
NCuts	0.72	0.82	2.7
DM + Kernel K-means	0.68	0.75	25.8
DM + Spectral Clustering	0.71	0.82	22.4
DM + NCuts	0.71	0.83	25.3
SSO + Kernel K-means	0.73	0.83	19.9
SSO + Spectral Clustering	0.74	0.84	24.7
SSO + NCuts	0.73	0.89	28.8
SD + Kernel K-means	0.75	0.87	5.4
SD + Spectral Clustering	0.77	0.93	6.3
SD + NCuts	0.82	0.97	6.0

single-light source images of 10 subjects seen under 585 viewing conditions (9 poses$\times$ 65 illumination conditions). For each pose, an image with ambient illumination was also captured. Hence, the total number of the images is $9 * 65 * 10 = 5,850$. We extract a subset of the dataset which consists of 3 out of 10 subjects. Therefore, the subset has $9 * 65 * 3 = 1,755$ faces.

We compare our method with some conventional clustering techniques: Kernel K-means [25], spectral clustering [35], Normalized Cuts [26]. Also, some recent affinity learning methods are compared : Diffusion Maps (DM), and Self-Smoothing Operator (SSO). These two methods iteratively learn a better affinity. Here we report: (1) F-score which is the harmonic mean of precision and recall, (2) the Normalized-Mutual-Information [28], and (3) CPU time that is used to perform both affinity learning and clustering. Every clustering method is run for 10 times independently, and the average scores are reported in Table.(1). We see that our method greatly improves the affinity that is used for kernel k-means, spectral clustering, and Normalized Cuts. The self-diffusion process accumulates the similarity mass among the intrinsic clusters. Note that our fast self-diffusion process is much more efficient than SSO since SSO iterates for at least hundreds of rounds to reach a promising stage $\mathcal{O}(tn^3)$, where t indicates the number of iterations and n is the size of similarity graph.

5.1.2 3D Image Clustering

We further test the proposed method for 3D image clustering. A benchmark dataset [22], COIL-20, is adopted, which consists of $1,420$ 3D images with 71 different illuminations and angles. The ground truth cluster number is 20. We compare the proposed method with with the affinity learning techniques: DM and SSO built around Kernel K-means, spectral clustering, and Normalized Cuts. The results are shown in Table(2): Our self-diffusion process gives about 10% improvement over the competing approaches, which is quite significant.

Table 2: A quantitative comparison on the COIL-20 dataset.

Methods/Score	F1 ↑	NMI ↑	CPU(s) ↓
Kernel K-means	0.62	0.68	1.8
Spectral Clustering	0.64	0.71	2.1
NCuts	0.68	0.77	2.3
DM + Kernel K-means	0.64	0.72	17.5
DM + Spectral Clustering	0.66	0.72	18.4
DM + NCuts	0.65	0.75	18.8
SSO + Kernel K-means	0.68	0.75	21.8
SSO + Spectral Clustering	0.70	0.77	22.4
SSO + NCut	0.73	0.79	23.3
SD + Kernel K-means	0.72	0.79	4.8
SD + Spectral Clustering	0.75	0.82	5.3
SD + NCuts	0.80	0.89	5.4

5.2. Segmentation

In a graph-based approach, the graph affinities are important for having a high quality segmentation result. We simply combine boundary and color cues of individual image pixels [7].

5.2.1 Unsupervised Image Segmentation

We first evaluate SD using the Berkeley image dataset [20]. There are 300 images in this dataset with each one having a ground truth label map. For a quantitative evaluation, four different measures are adopted: 1) Probabilistic Rand Index(PRI) [33], which counts the number of pixel pairs whose labels are consistent between the segmentation and the ground truth; 2) Variation of Information (VoI) [21], which measures the amount of randomness in one segmentation that cannot be explained by the other; 3)Global Consistency Error (GCE) [20], which measures the extent to which one segmentation can be viewed as a refinement of the other and 4) Boundary Displacement Error (BDE) [13], which measures the average displacement error of the boundary pixels between two segmented images.

We compare our method with standard methods: Mean Shift [6], NCuts [26], JSEG [8], Graph-based Segmentation [11], MNCut [7], Normalized Tree Partitioning(NTP) [34], Saliency-based Segmentation(Saliency) [9], and Full Affinity Cut (FAC) [17]. Note that DM and SSO are not compared with due to the computational reasons. Quantitative comparison is shown in Table(3). Note that Full Affinity Cut(FAC) [17] achieves a slightly better result than our method since they use over-segmentation to generate regions to guide the segmentation process; in our case, only pixel-level cues are used.

To give a full comparison with spectral methods, e.g.,

Table 3: A quantitative comparison on the Berkeley dataset [20].

Methods/Score	PRI ↑	VoI ↓	GCE ↓	BDE ↓
MShift	0.7958	1.9725	0.1888	14.41
NCuts	0.7242	2.9061	0.2232	17.15
JSEG	0.7756	2.3217	0.1989	14.40
GBIS	0.7139	3.3949	0.1746	16.67
MNCut	0.7559	2.4701	0.1925	15.10
NTP	0.7521	2.4954	0.2373	16.30
Saliency	0.7758	1.8465	0.1768	16.24
FAC	0.8146	1.8545	0.1809	12.21
Our algorithm	0.8037	1.8458	0.1802	13.87

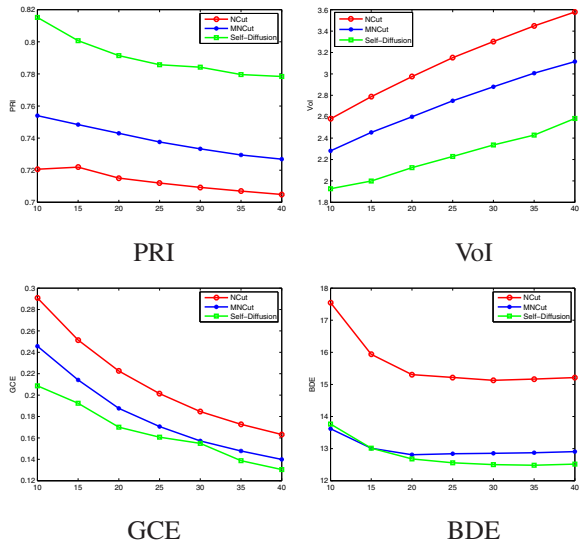


Figure 5: A quantitative comparison with spectral methods on different segment numbers.

NCut and MNCut, the proposed methods are tested with different segment number. Fig.(5) shows the four quantities with different number of segments. These comparisons present that our algorithm outperforms NCut and MNCut in all cases. Another common benchmark, MSRC dataset [27] is used to test our method. We only give qualitative comparison with NCut and MNCut in Fig.(6). Compared with two spectral methods: NCuts and MNCut, our algorithm produces perceptually and quantitatively higher quality segmentations. Segmenting one image using self-diffusion takes from ten seconds to two minutes on a Pentium 1.7 GHz for the dataset.

5.2.2 Interactive Segmentation

Now we demonstrate the effectiveness of our method on interactive segmentation (graph-based Laplacian for semi-supervised learning). The proposed algorithm is tested on the Microsoft GrabCut database which is composed of 50 images with tri-maps and ground truth segmentations. This database is widely used for quantitative segmentation com-

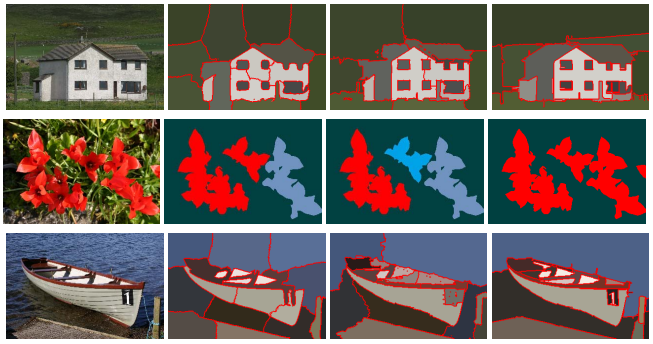


Figure 6: A qualitative comparison with spectral methods in the MSRC dataset[27].

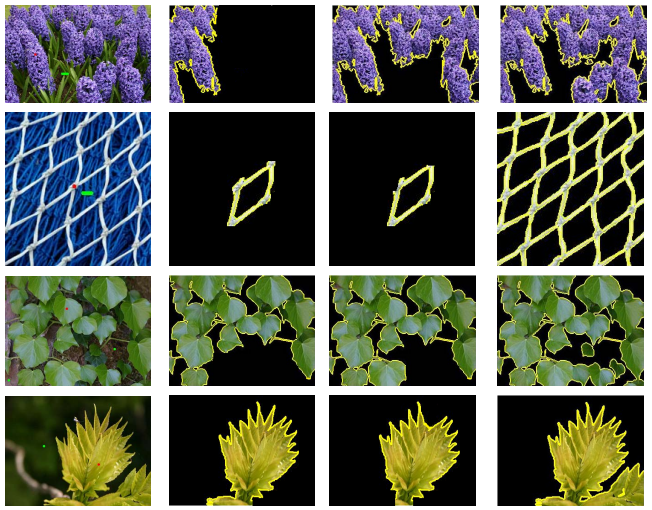


Figure 7: A qualitative comparison with random walk (second column) and grab cut (third column) in interactive segmentation. The last column shows the result of our method.

Table 4: A comparison of error rate on the Microsoft GrabCut database.

Methods	GMMRF	P^n Model	Random Walk	GraphCut	SD + GCut
Error	7.9%	6.08%	5.6%	5.4%	4.6%

parison since it also provides information about seed regions. Although most pixels are labeled as the seeds except for a narrow band around the objects, this dataset provides a common ground for an objective evaluation of different semi-automatic segmentation algorithms. The proposed method is compared with four popular algorithms for interactive segmentations: GMMRF [4], Robust P^n model [19], Random Walk [15] and GraphCut [23]. To further highlight the differences, we use a more comprehensive image dataset [18] in which all the images are provided with user-defined labels indicated by scribbles of red and green labels (See Fig. 7). Some examples of segmentation results in Fig.(7).

6. Conclusion

In this paper, we have introduced a new algorithm, self-diffusion, for learning affinity map. Theoretical analysis is given and a proper approximation of the optimal stoping time t^* is provided. Furthermore, we develop a fast self-diffusion process to make it applicable to practical image clustering and segmentation.

Acknowledgment: This work is supported by Office of Naval Research Award N000140910099 and NSF CAREER award IIS-0844566.

References

[1] X. Bai, X. Yang, L. Latecki, W. Liu, and Z. Tu. Learning context sensitive shape similarity by graph transduction. *IEEE Trans. PAMI*, 2010.

[2] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *J. of Machine Learning Research*, 7:2399–2434, 2006.

[3] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. PAMI*, 24:705–522, 2002.

[4] A. Blake, C. Rother, M. Brown, P. Perez, and P. Torr. Interactive image segmentation using an adaptive gmmrf model. In *in ECCV*, 2004.

[5] R. Coifman and S. Lafon. Diffusion maps. *Applied and Comp. Harmonic Ana.*, 2006.

[6] D. Comaniciu, P. Meer, and S. Member. Mean shift: A robust approach toward feature space analysis. *IEEE Tran. on PAMI*, 24:603–619, 2002.

[7] T. Cour, F. Benezit, and J. Shi. Spectral segmentation with multiscale graph decomposition. In *Proc. of CVPR*, pages 1124–1131, 2005.

[8] Y. Deng and B. S. Manjunath. Unsupervised Segmentation of Color-Texture Regions in Images and Video. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(8):800–810, 2001.

[9] M. Donoser, M. Urschler, M. Hirzer, and H. Bischof. Saliency driven total variation segmentation. 2009.

[10] O. Duchenne, J.-Y. Audibert, R. Keriven, J. Ponce, and F. Segonne. Efficient planar graph cuts with applications in computer vision. In *Proc. of CVPR*, 2008.

[11] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59:2004, 2004.

[12] F. Fouss, A. Pirotte, J. michel Renders, and M. Saerens. Random-walk computation of similarities between nodes of a graph, with application to collaborative recommendation. *IEEE Tran. on Know. and Data Eng.*, 19:2007, 2006.

[13] J. Freixenet, X. Mu?oz, D. Raba, J. Martł, and X. Cufł. Yet another survey on image segmentation: Region and boundary information integration. In *In ECCV*, pages 408–422, 2002.

[14] A. Georghiades, P. Belhumeur, and D. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Tran. on PAMI*, 23(6):643–660, 2001.

[15] L. Grady. Random Walks for Image Segmentation. *IEEE Tran. on PAMI*, 28(11):1768–1783, 2006.

[16] J. Jiang, B. Wang, and Z. Tu. Unsupervised metric learning by self-smoothing operator. In *ICCV*, 2011.

[17] T. H. Kim and K. M. Lee. Learning full pairwise affinities for spectral segmentation. *CVPR*, pages 2101–2108, 2010.

[18] T. H. Kim, K. M. Lee, and S. U. Lee. Nonparametric higher-order learning for interactive segmentation. *CVPR*, pages 3201–3208, 2010.

[19] P. Kohli and P. H. S. Torr. Robust higher order potentials for enforcing label consistency. In *In CVPR*, 2008.

[20] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *ICCV*.

[21] M. Meila. Comparing clusterings: an axiomatic view. In *In Proceedings of the 22nd international conference on Machine learning*, pages 577–584. ACM Press, 2005.

[22] S. A. Nene, S. K. Nayar, and H. Murase. Columbia Object Image Library (COIL-20). Technical report, Feb 1996.

[23] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM SIGGRAPH*, 2004.

[24] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[25] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[26] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. on PAMI*, 22:888–905, 1997.

[27] J. Shotton, M. Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *CVPR*, 2008.

[28] A. Strehl, J. Ghosh, and C. Cardie. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617, 2002.

[29] M. Szummer and T. Jaakkola. Partially labeled classification with markov random walks. In *NIPS*, pages 945–952, 2001.

[30] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500), 2000.

[31] N. Tishby and N. Slonim. Data clustering by markovian relaxation and the information bottleneck method. In *Proc. of NIPS*, 2000.

[32] Z. Tu and S. Zhu. Image segmentation by data-driven markov chain monte carlo. *IEEE Trans. on PAMI*, 2002.

[33] R. Unnikrishnan, C. Pantofaru, and M. Hebert. Toward Objective Evaluation of Image Segmentation Algorithms. *IEEE Tran. on PAMI*, 29(6):929–944, 2007.

[34] J. Wang, Y. Jia, X.-S. Hua, C. Zhang, and L. Quan. Normalized tree partitioning for image segmentation. *CVPR*, pages 1–8, 2008.

[35] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *NIPS*, 2004.

[36] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In *Proc. NIPS*, 2004.

[37] X. Zhu. Semi-supervised learning with graphs. In *Doctoral Dissertation, CMU-LTI-05-192*, 2005.

[38] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, 2003.

7. Appendix

Derivation of Theorem 4.4

$$\begin{aligned}
& \nabla DoF(W_t) = Trace(W_{t+2} - 2W_{t+1} + W_t) \\
= \;& Trace((W_tP + I)P + I - 2(W_tP + I) + W_t) \\
= \;& Trace(W_t(I - P)^2 + (P - I)) \\
= \;& Trace(((I - P^t)(I - P)^{-1} + WP^t)(I - P)^2 + P - I) \\
= \;& Trace((W - WP - I)(P^t - P^{t+1})) \\
\leq \;& |(W - WP - I)|_2 Trace(P^t - P^{t+1}) \qquad (17)
\end{aligned}$$

Note that we repeatedly use the following two formula about traces of matrices: $trace(AB) = trace(BA)$. Also, we used the property that $W - WP - I$ is symmetric(Hermitian). From Eqn.(17), we can find that when $P^t \approx P^{t+1}$, i.e., the random walks start to lose discriminative power, Self-Diffusion achieve the best accumulative power. Let $\{\lambda_i, i = 1, \ldots, n\}$ be the eigenvalues of transition matrix P, we have $1 = \lambda_1 > \lambda_2 > \ldots, > \lambda_n$. Then, if we restrict t be an even number,

$$\begin{aligned}
Trace(P^t - P^{t+1}) &= \sum_i^n (\lambda_i^t - \lambda_i^{t+1}) \\
&\leq \lambda_2^t \sum_i^n (1 - \lambda_i) \\
&= \lambda_2^t (n - Trace(P)) \qquad (18)
\end{aligned}$$

Combining (17) and (18), we have $t^* = \lceil \ln(\frac{\epsilon}{(n - DoF(P))\|(W - WP - I\|)} / \ln(\lambda_2) \rceil$. Since λ_2 is very close to $\lambda_1 = 1$, to ensure numerical robustness of the estimator t^*, we add a parameter C to λ_2:

$$t^* = \lceil \ln(\frac{\epsilon}{(n - DoF(P))\|(W - WP - I\|)} / \ln(\lambda_2 - C) \rceil$$

. Empirical experience shows that when $C \in [0.01, 0.1]$, the estimator of t^* is most effective.