$\mathcal{G}raph\mathcal{T}_7$

DIFFUSION ON GRAPHS

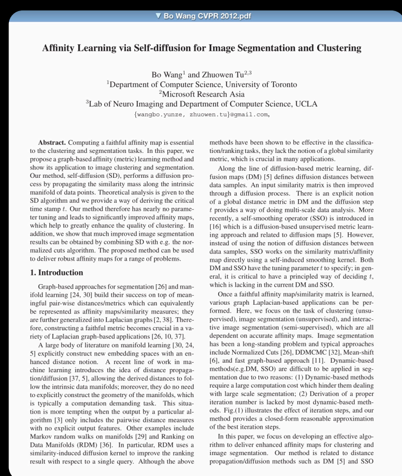
Information flow efficiency [GT 6]

machine Learning & pattern recognition

segment images

diffusion on graphs

[Wang et al. CVPR 2012]


▼ Bo Wang CVPR 2012.pdf

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.

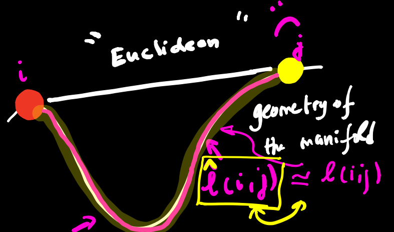
Once a faithful affinity map/similarity matrix is learned, various graph Laplacian-based applications can be performed. Here, we focus on the task of clustering (unsupervised), image segmentation (unsupervised), and interactive image segmentation (semi-supervised), which are all dependent on accurate 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

Q1 What is the problem to solve?

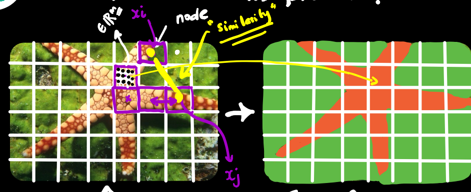
Given an input image — label map

automatic segmentation [labeling



learn how to estimate the length of this path.

Q2 How to model this problem?



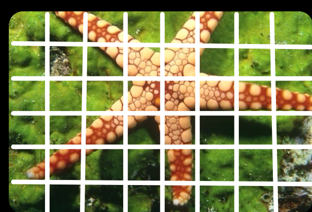
Original image we want to segment

desired output: segmentation map / labeled image

How to leverage graphs to solve this problem?

Solution: learn the similarity (i.e., edge weights) between pairs of patches (i.e., nodes) in the input image (i.e., graph).

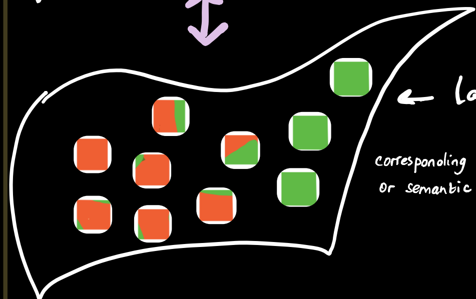
Why not use Euclidean distance between two "samples" (e.g. patches or pixels)?



- Euclidean distance does not capture the geometry of a high-dimensional manifold where the data samples live.

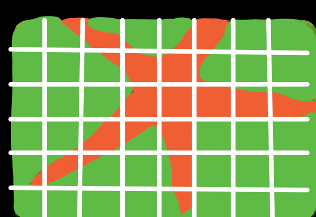
image manifold ∈ ℝ^m

← image



← label

corresponding labeled patch or semantic patch



Q3 How to solve this problem?

Learning sample similarity / affinity for labeling the input image

W = affinity / similarity / (metric) matrix

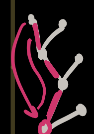
Proposed solution → Wang et al. propose a graph-based affinity (metric) learning method based on diffusion.

learning an accurate / faithful affinity map / similarity matrix affects the segmentation accuracy.

* once W is properly learned one can use a conventional segmentation method to partition the image into semantically homogeneous regions.

How to estimate W?

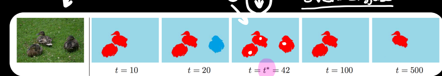
- Use graph self-diffusion (SD) → local / global



It works directly on the input affinity map and uses regularization in each diffusion iteration.

↳ assumption | long-range similarities can be approximated by an accumulation of local similarities.

" over-diffuse "



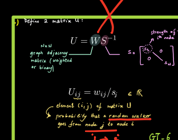
" diffusion iteration " →

- Given a graph $G = (\Omega, W)$ where $\Omega = \{x_i\}_{i=1}^{n}$ is a collection of data samples.
- W is the similarity matrix in $\mathbb{R}^{n \times n}$.
- $W(i,j) \in [0,1]$, where $W(i,j) = 1 \Leftrightarrow x_i = x_j$.

1) $W(i,j) = \overbrace{exp(-d^2(i,j)/\sigma^2)}^{\text{Gaussian kernel}}$ — d: Euclidean distance; σ: kernel size / " scale "

2) $D(i,i) = \sum_{k=1}^{n} W(i,k)$ (diagonal node strength matrix)

3) $P = D^{-1}W$ (transition kernel → row-wise normalization of W similarity matrix)



$P_{ij} = \frac{w_{ij}}{s_i}$, $i \to j$

$W = \begin{bmatrix} w_{11} & & \\ w_{21} & w_{22} \cdots & \cdots \\ w_{ij} & & \end{bmatrix}$ / s_i

4) $W_t = W_{t-1}P + I$ (I: regularization) → iterative self-diffusion of W

5) $W^* = W_{t^*}D^{-1}$ (closed form solution) — W^*: final learned affinity matrix; t^*: optimal number of iterations

$\boxed{t^* = \ ?}$

Why it works? [theoretical proof]

Intuitive explanation is good but theoretical proof is best!

↳ There is a closed form for estimating an appropriate t^* at which the effect of SD is best.

1. Computing the smoothing kernel:

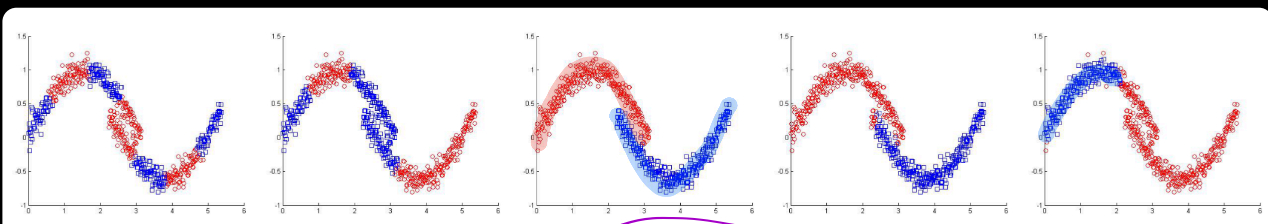
$$P = D^{-1}W$$

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

2. Performing smoothing for t^* steps: Wang et al 2012

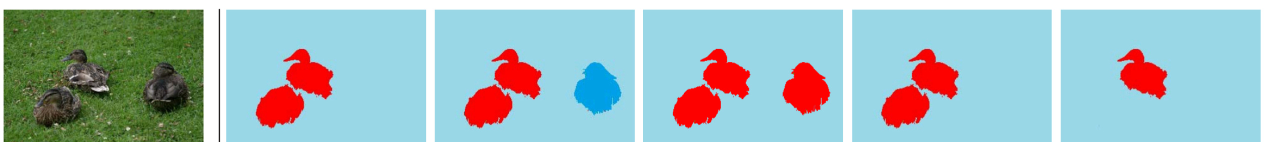
$$W_t = W_{t-1}P + I.$$

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

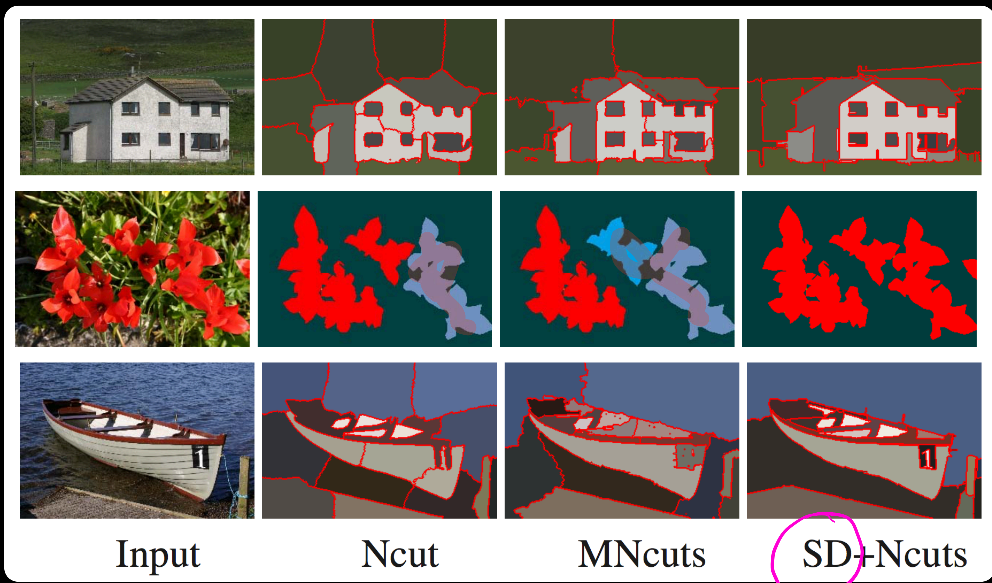


$t = 10$ | $t = 30$ | $t = t^* = 52$ | $t = 80$ | $t = 150$

$t > t^*$



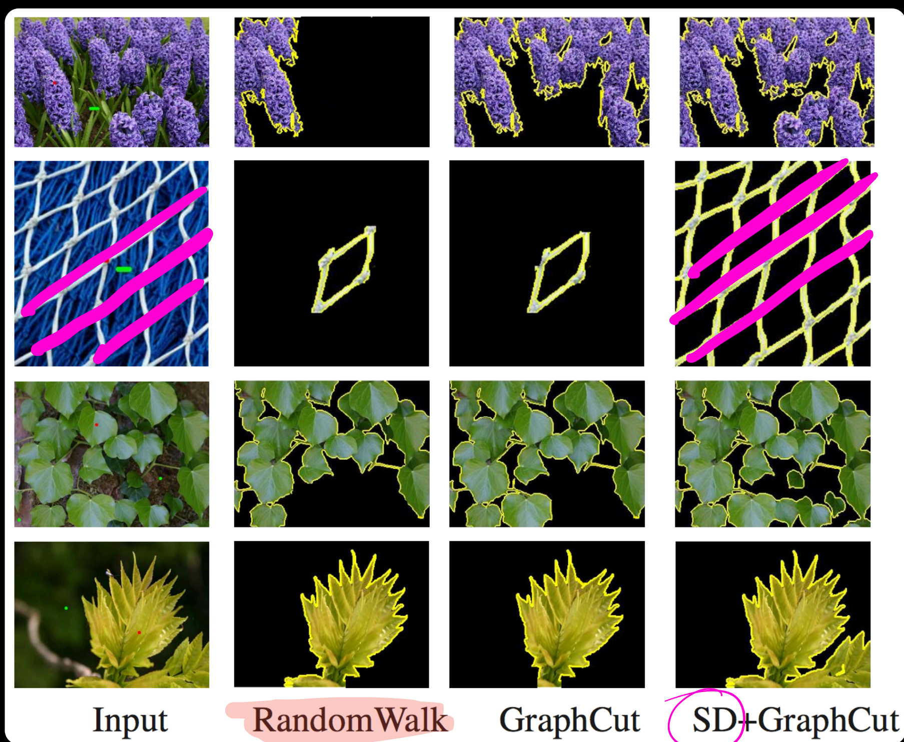
$t = 10$ | $t = 20$ | $t = t^* = 42$ | $t = 100$ | $t = 500$



normalized cut

multi-scale normalized cuts

W^*



$$\begin{cases} P = D^{-1} W \\ W_t = W P^t \end{cases}$$

running random walk t times

Bo Wang CVPR 2012.pdf

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

input image

Normalized Cut

Normalized Cut + SD

scikit-image
image processing in python

Download Gallery Documentation Community Guidelines Source

Search documentation ...

Docs for 0.17.dev0
All versions

Note
Click here to download the full example code

Normalized Cut

This example constructs a Region Adjacency Graph (RAG) and recursively performs a Normalized Cut on it 1.

References

1 Shi, J.; Malik, J., "Normalized cuts and image segmentation", Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 22, no. 8, pp. 888-905, August 2000.

```
from skimage import data, segmentation, color
from skimage.future import graph
from matplotlib import pyplot as plt


img = data.coffee()

labels1 = segmentation.slic(img, compactness=30, n_segments=40
0)
out1 = color.label2rgb(labels1, img, kind='avg')

g = graph.rag_mean_color(img, labels1, mode='similarity')
labels2 = graph.cut_normalized(labels1, g)
out2 = color.label2rgb(labels2, img, kind='avg')

fig, ax = plt.subplots(nrows=2, sharex=True, sharey=True, figs
ize=(6, 8))

ax[0].imshow(out1)
ax[1].imshow(out2)

for a in ax:
    a.axis('off')

plt.tight_layout()
```

Total running time of the script: (0 minutes 5.416 seconds)

Download Python source code: plot_ncut.py

Download Jupyter notebook: plot_ncut.ipynb

← Segmentation
" $\underline{\underline{W}}$ "
step 1
$\underline{\underline{W^*}}$ →

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: $W^* = W_t * D^{-1}$