Semi-Supervised Shape Classification with Manifold Regularization

Stanislav Nikolov SNIKOLOV@MIT.EDU

Abstract

We approach the problem of semi-supervised shape classification by exploiting the geometric structure of shape data. We apply *manifold regularization* to learn a function from shapes to class labels. Central to manifold regularization algorithms is the use of a weighted graph to represent pairwise relationships between training points and capture their geometric structure. Under a regularized least squares formulation, each algorithm only involves solving a linear system of equations. We analyze the classification performance for different features regularization parameters, percentage of labeled data, and noise in the labels. We show that encouraging the smoothness of the classification function on the manifold improves classification performance beyond simply encouraging smoothness in the ambient space. We compare raw pixel features to Signed Distance Function (SDF) features and find that SDF features are yield consistently higher accuracy. Finally, we compare manifold regularization to the much simpler k-Nearest-Neighbors and show that manifold regularization is consistently better.

Keywords: shape classification, manifold regularization, semi-supervised learning.

1. Introduction

In many object recognition and classification tasks, the shape of an object is more important than its other qualities, such as color or texture. For example consider the task of classifying images of bottles and cups. The bottles and cups themselves might have a variety of colors and textures, in addition to noise and variation in background and lighting. In comparison, the silhouettes of cups and bottles have fewer, but more important degrees of freedom. Shape classification aims to exploit this.

Shape classification has many applications. Crowdsourced image annotation tools such as LabelMe Russell et al. (2008) have generated hundreds of thousands of labeled shapes from segmentions of everyday scenes, making it as feasible as ever to learn models for everyday shapes and exploit them to do object recognition in complex scenes. In biomedical imaging, better classification of segmented shapes could help with early detection of disease.

Recently, geometric frameworks for learning such as manifold learning and manifold regularization (Belkin et al. (2005), Belkin et al. (2004), Belkin et al. (2006)) have become a popular approach to image and shape classification (Chan and Zhu (2005), Tuzel et al. (2008), Zhu et al. (2009), Gong et al. (2008)). It is well known that the space of shapes has the structure of a low-dimensional manifold (see for example Haykin (2007), Chapter 1. The idea is that the raw pixels of an image form a feature vector in a high-dimensional space, but around each feature vector, the possible variation in shape happens only in a

small number of directions. Roughly, the low dimension of the manifold corresponds to a low number of latent degrees of freedom in the set of shapes.

2. Manifold Regularization

In semi-supervised learning, there is a set of n points $\{\mathbf{x}_i\}_{i=1,\dots,n}$, of which the first ℓ have labels $\{y_i\}_{i=1,\dots,\ell}$ and the remaining $u = n-\ell$ are unlabeled. The goal is to learn a function f that predicts the label of an arbitrary (potentially unobserved) point. In a fully-supervised setting, one would use only the labeled points to learn f. Regularized Least Squares (RLS), for example, attempts to learn f by minimizing the regularized squared loss

$$\frac{1}{\ell}\sum_{i=1}^{\ell}(f(\mathbf{x}_i) - y_i)^2 + \lambda_A \|f\|_{\mathcal{H}}^2$$

which can be written as

$$\frac{1}{\ell}\|\mathbf{K}\mathbf{c} - \mathbf{y}\|_2^2 + \lambda_A \mathbf{c}^T\mathbf{K}\mathbf{c}$$

using the kernel matrix $\mathbf{K}$ and coefficients $\mathbf{c}$ given by the Representer Theorem. Its solution is given by taking the derivative of the objective with respect to $\mathbf{c}$ and setting it to zero. This yields

$$-\frac{\mathbf{y}}{\ell} + \left(\frac{\mathbf{K}}{\ell} + \lambda_A \mathbf{I}\right)\mathbf{c}^* = 0 \tag{1}$$

$$\mathbf{c}^* = (\mathbf{K} + \ell\lambda_A \mathbf{I})^{-1}\mathbf{y} \tag{2}$$

Manifold regularization uses the unlabeled points in addition to the labeled points in order to exploit the geometry of the data (Belkin et al. (2005), Belkin et al. (2004), Belkin et al. (2006)). Namely, it enforces the notion that f should not vary drastically in regions of high density. This idea is captured formally by defining an intrinsic smoothness penalty on f

$$\|f\|_I^2 = \int_{\mathcal{M}} \|\nabla_{\mathcal{M}} f(x)\|^2 dp(\mathbf{x}) = \int \Delta_{\mathcal{M}} f dp(\mathbf{x}),$$

where $\nabla_{\mathcal{M}}$ and $\Delta_{\mathcal{M}}$ are the gradient and Laplacian of f on the manifold, respectively. This gives rise to Laplacian Regularized Least Squares (LapRLS) where one minimizes

$$\frac{1}{\ell}\sum_{i=1}^{\ell}(f(\mathbf{x}_i) - y_i)^2 + \lambda_A\|f\|_{\mathcal{H}}^2 + \lambda_I\|f\|_I^2.$$

Furthermore, one can approximate the intrinsic smoothness penalty using only the observed points by using a graph Laplacian instead of the manifold Laplacian: $\mathbf{x}_1, \dots, \mathbf{x}_n$:

$$\|f\|_I^2 \approx \frac{1}{2n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} W_{i,j}(f_i - f_j)^2 = \mathbf{f}^T\mathbf{L}\mathbf{f}.$$

The weights

$$W_{i,j} = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}}$$

are similarity weights for some scale parameter σ and $\mathbf{L}$ is the graph Laplacian of the graph defined by the $W_{i,j}$. Manifold regularization has its own representer theorem, which allows us to write the objective as

$$\frac{1}{\ell}\|\mathbf{J}(\mathbf{Kc}-\mathbf{y}')\|^2+\lambda_A\mathbf{c}^T\mathbf{Kc}+\frac{\lambda_I}{n^2}\mathbf{c}^T\mathbf{KLKc}$$

where $\mathbf{J}$ is n by n with ones in the first ℓ positions along the diagonal and zeros everywhere else and $\mathbf{y}'$ is a vector of length n with the labels of the labeled points in the first ℓ values and zeros everywhere else. Taking the derivative of the objective with respect to $\mathbf{c}$ and setting it to zero, we get

$$\frac{1}{\ell}\mathbf{KJ}\left(\mathbf{y}'-\mathbf{JKc}^*\right)+\left(\lambda_A\mathbf{K}+\frac{\lambda_I\ell}{n^2}\mathbf{KLK}\right)\mathbf{c}^*=0$$

Solving for the optimal coefficients $\mathbf{c}^*$ yields

$$\mathbf{c}^*=\left(\mathbf{JK}+\lambda_A\ell\mathbf{I}+\frac{\lambda_I\ell^2}{n^2}\mathbf{LK}\right)^{-1}\mathbf{y}'. \tag{3}$$

To minimize the regularized loss, we simply have to solve a linear system.

3. Data

To represent shapes, we used binary "silhouette" images from the Large Binary Image Database (`http://www.lems.brown.edu/~dmc/`). We randomly sampled a roughly equal amount of images of butterflies (161), dogs (154), heads (149), and fish (156). The images

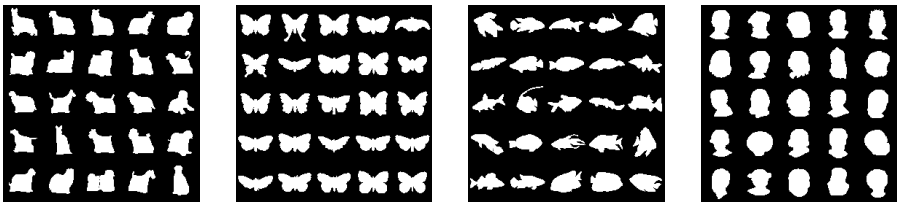

Figure 1: Dog, butterfly, fish and human head shapes as binary silhouette images.

are not all the same dimensions and are typically rectangular. Therefore, via padding and cropping, we converted each image to be s by s pixels, where s is the larger of the maximum width and maximum height over all images. As an optional step, one can rescale each image so that that shapes are of comparable size. We found most of the shapes to be of similar scale and therefore omitted this step. Next, we created two feature representations from each image, each of which is "image-like" and has the same dimensions as the original image. The first feature representation is simply the raw pixels of the image. The second feature representation is the discretized *signed distance function*, which we will describe shortly. Finally, to reduce dimensionality, each feature representation is scaled down.

3.1. Signed Distance Functions

A *signed distance function* (SDF) is an object that implicitly defines a partition of a space into (possibly multiply-connected) inside and outside regions. Signed distance functions are widely used in computer vision to represent shapes (see for example Chan and Zhu (2005), Tsai et al. (2003), Tsai et al. (2005)). In our case, the space is a plane, and the boundary between the inside and outside regions can be thought of as a shape. SDFs take in a point $\mathbf{x}$ (e.g. in the plane) and give the *signed distance* to the closest point on the boundary. If $\mathbf{x}$ is in the inside region, the signed distance is negative; if it is in the outside region, the signed distance is positive. Hence, the boundary, or shape, is given by the zero contour of the SDF.

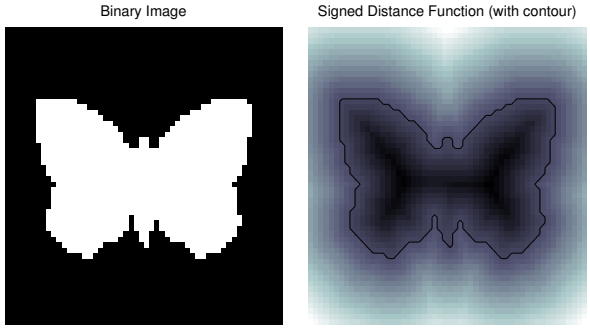

Figure 2: Signed Distance Functions are implicit representations of shapes.

4. Experiment

4.1. Goals

Our goal is to study three things: the effect of having an intrinsic regularization parameter λ_I, the effect of using SDF features over raw image features, and how the performance of manifold regularization (MR) compares to that of k-Nearest-Neighbors (kNN).

4.2. Training/Test Split

For each pair of datasets, we form a test set using 25% of the data and use the remaining 75% to train a classifier. We reveal the labels of a fraction $p_{labeled}$ of the training set and leave the rest unlabeled. Of those labeled, we artificially add noise by flipping each label with probability p_{flip}. We then calculate the percent correct on the test set. Figure 3 illustrates this setup visually.

4.3. Cross-Validation and Parameter Tuning

During training, certain parameters are automatically tuned for each algorithm by minimizing the k-fold cross validation error (k=8). We use only the labeled points in the test portion of each validation split. That is, the points originally chosen to be unlabeled are still

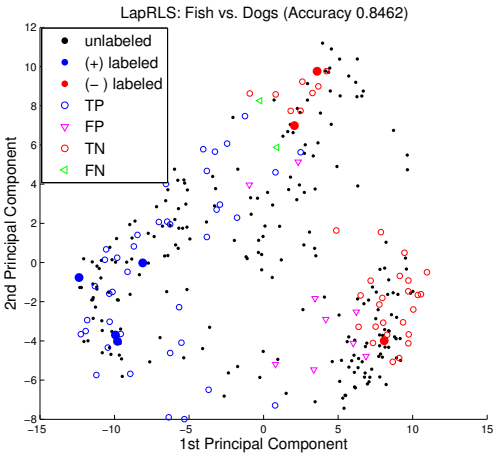

Figure 3: An illustration of classification, showing unlabeled data, labeled data, and classified test data. To visualize the high dimensional features, we plotted the data along its first two principal component.

considered unlabeled and are not used for computing cross-validation error. For manifold regularization, the parameters to be tuned are λ_A for RLS, λ_A and λ_I for LapRLS, and σ for both RLS and LapRLS. For kNN, the parameter to be tuned is simply k.

The range for each parameter to be tuned was chosen by trial and error on the datasets involved. In particular, we made sure that the optimal value of the parameter did not often lie at either end of the range searched. For the k in kNN we search from $k = 1, \ldots, 10$. For some parameters, especially low $p_{labeled}$, the number of labeled points in the training part of a validation split does not allow for all values of k in this range. For example, if $k = 10$ and the number of labeled points is less than 10, there cannot be 10 nearest neighbors. In this case, we search only through the allowed values of k. For σ, we compute the average distance between points and search a range of multiples from 0.5 to 1.5 of this value. The average distance defines the typical length scale to be used when computing graph weights between points. For the regularization parameters λ_A and λ_I, we search linearly from 0 to $\lambda_{\max}(\mathbf{K})$. Roughly speaking, we would like to avoid an ill-conditioned system in Eqs. 3 and 2, and regularization parameters on the order of magnitude of the maximum eigenvalue of $\mathbf{K}$ make this possible.

For the kernel $\mathbf{K}$, we chose the same form as the weight matrix $\mathbf{W}$ with σ equal to a multiplicative factor of 1 times the average distance between all points. For simplicity, we fixed the multiplicative factor to 1 after initial trial and error, and focus instead on the effects of the other parameters.

4.4. Multiple Trials

The test set performance is computed for 5 different trials, with the random seed being reset before each set of trials. This way, the same random choices of points are involved, and we can better gauge the effect of just changes in parameters.

4.5. Mild and Harsh Conditions

Each algorithm is evaluated under "harsh" conditions and "mild" conditions. Under mild conditions ($p_{labeled} = 0.25$, $p_{flip} = 0$), 25% of the training set is labeled and the labels are noiseless. Under harsh conditions ($p_{labeled} = 0.05$, $p_{flip} = 0.2$), only 5% of the training set is labeled and the labels are noisy.

4.6. Algorithm Comparison

To study each effect mentioned in section 4.1, we propose the following methodology.

To compare kNN and MR, we compare the results of kNN to the best results from MR over all choices of regularization mode (RLS vs. LapRLS) and all choices of features.

To compare raw image features and SDF features, we compare, for each pair of datasets, the best performing MR algorithm (either RLS or LapRLS) that uses image features, and the best-performing MR algorithm that uses SDF features.

Finally, to compare RLS and LapRLS, we directly compare their accuracy for each pair of datasets. Note that each of these comparisons is done for each pair of datasets under both mild and harsh conditions.

5. Results

5.1. Effect of Regularization

Table 1 shows that LapRLS consistently has higher accuracy than RLS.

5.2. Comparison of Features for Manifold Regularization

We took the better of each pair of LapRLS and RLS results from Table 1 and compared the classification performance of raw image feature and SDF features. Table 2 shows that SDF features do better than image features.

5.3. Comparison of Features for kNN

In order to compare kNN to manifold regularization, we need to take the better of each pair of SDF and raw image kNN results. Table 3 shows the accuracy of kNN for each type of feature. It appears that neither feature is clearly better than the other.

5.4. Comparison of Manifold Regularization and kNN

We compared the best results from manifold regularization to the best results from kNN. We took the better of each pair of image and SDF feature results from Table 2 for manifold regularization, and from Table 3 for kNN. Table 4 shows that manifold regulariation has higher accuracy.

$p_{flip} = 0.2$, $p_{labeled} = 0.05$					$p_{flip} = 0$, $p_{labeled} = 0.25$				
Image features					Image features				
Class 1	Class 2	Features	Mode	Accuracy	Class 1	Class 2	Features	Mode	Accuracy
butterflies	fish	image	RLS	0.58±0.07	butterflies	fish	image	RLS	0.77±0.09
butterflies	fish	image	LapRLS	**0.82±0.04**	butterflies	fish	image	LapRLS	**0.90±0.02**
butterflies	heads	image	RLS	0.80±0.22	butterflies	heads	image	RLS	**0.99±0.01**
butterflies	heads	image	LapRLS	**0.81±0.23**	butterflies	heads	image	LapRLS	0.99±0.02
butterflies	dogs	image	RLS	0.62±0.17	butterflies	dogs	image	RLS	0.87±0.10
butterflies	dogs	image	RLS	**0.76±0.10**	butterflies	dogs	image	LapRLS	**0.95±0.01**
fish	heads	image	RLS	0.75±0.16	fish	heads	image	RLS	0.95±0.05
fish	heads	image	LapRLS	**0.95±0.03**	fish	heads	image	LapRLS	**0.99±0.00**
fish	dogs	image	RLS	0.61±0.14	fish	dogs	image	RLS	0.83±0.02
fish	dogs	image	LapRLS	**0.69±0.16**	fish	dogs	image	LapRLS	**0.90±0.10**
heads	dogs	image	RLS	0.58±0.14	heads	dogs	image	RLS	0.85±0.12
heads	dogs	image	LapRLS	**0.75±0.12**	heads	dogs	image	LapRLS	**0.94±0.04**
SDF features					SDF features				
butterflies	fish	SDF	RLS	0.77±0.03	butterflies	fish	SDF	RLS	0.93±0.01
butterflies	fish	SDF	LapRLS	**0.86±0.10**	butterflies	fish	SDF	LapRLS	**0.94±0.02**
butterflies	heads	SDF	RLS	0.98±0.03	butterflies	heads	SDF	RLS	**1.00±0.00**
butterflies	heads	SDF	LapRLS	**0.99±0.00**	butterflies	heads	SDF	LapRLS	**1.00±0.00**
butterflies	dogs	SDF	RLS	**0.91±0.04**	butterflies	dogs	SDF	RLS	0.98±0.01
butterflies	dogs	SDF	LapRLS	0.91±0.06	butterflies	dogs	SDF	LapRLS	**0.98±0.00**
fish	heads	SDF	RLS	0.97±0.01	fish	heads	SDF	RLS	0.98±0.01
fish	heads	SDF	LapRLS	**0.98±0.01**	fish	heads	SDF	LapRLS	**0.99±0.01**
fish	dogs	SDF	RLS	0.87±0.02	fish	dogs	SDF	RLS	0.96±0.01
fish	dogs	SDF	LapRLS	**0.89±0.03**	fish	dogs	SDF	LapRLS	**0.97±0.01**
heads	dogs	SDF	RLS	0.73±0.22	heads	dogs	SDF	RLS	0.94±0.01
heads	dogs	SDF	LapRLS	**0.93±0.03**	heads	dogs	SDF	LapRLS	**0.99±0.01**

Table 1: LapRLS outperforms RLS for both image and SDF features, as well as under both noisy and sparsely labeled conditions ($p_{flip} = 0.2$, $p_{labeled} = 0.05$) and noiseless and more densely labeled conditions ($p_{flip} = 0$, $p_{labeled} = 0.25$).

$p_{flip} = 0.2$, $p_{labeled} = 0.05$				$p_{flip} = 0$, $p_{labeled} = 0.25$			
Class 1	Class 2	Features	Accuracy	Class 1	Class 2	Features	Accuracy
butterflies	fish	image	0.82±0.04	butterflies	fish	image	0.90±0.02
butterflies	fish	SDF	**0.86±0.10**	butterflies	fish	SDF	**0.94±0.02**
butterflies	heads	image	0.81±0.23	butterflies	heads	image	0.99±0.01
butterflies	heads	SDF	**0.99±0.00**	butterflies	heads	SDF	**1.00±0.00**
butterflies	dogs	image	0.76±0.10	butterflies	dogs	image	0.95±0.01
butterflies	dogs	SDF	**0.91±0.04**	butterflies	dogs	SDF	**0.98±0.00**
fish	heads	image	0.95±0.03	fish	heads	image	**0.99±0.00**
fish	heads	SDF	**0.98±0.01**	fish	heads	SDF	0.99±0.01
fish	dogs	image	0.69±0.16	fish	dogs	image	0.90±0.10
fish	dogs	SDF	**0.89±0.03**	fish	dogs	SDF	**0.97±0.01**
heads	dogs	image	0.75±0.12	heads	dogs	image	0.94±0.04
heads	dogs	SDF	**0.93±0.03**	heads	dogs	SDF	**0.99±0.01**

Table 2: The table above compares the best results for image features to the best results for SDFs for manifold regularization. The best result for a given pair of datasets and a choice of features is the maximum of the accuracies given by RLS and LapRLS (found in Table 1). It is clear that SDF features outperform image features.

6. Discussion and Summary

6.1. LapRLS vs. RLS

It is no surprise that LapRLS has better classification performance than RLS. Spaces of shapes are well-known to have a low-dimensional manifold structure, and only LapRLS is able to exploit it. For sufficiently mild conditions and when using the better-performing SDF features, accuracy is often near-perfect, and the classification performances of LapRLS

$p_{flip} = 0.2$, $p_{labeled} = 0.05$

Class 1	Class 2	Features	Accuracy
butterflies	fish	image	0.68±0.11
butterflies	fish	SDF	**0.69±0.12**
butterflies	heads	image	0.97±0.04
butterflies	heads	SDF	**0.97±0.01**
butterflies	dogs	image	**0.73±0.16**
butterflies	dogs	SDF	0.71±0.12
fish	heads	image	**0.73±0.40**
fish	heads	SDF	0.71±0.40
fish	dogs	image	**0.78±0.23**
fish	dogs	SDF	0.73±0.25
heads	dogs	image	**0.75±0.22**
heads	dogs	SDF	0.74±0.21

$p_{flip} = 0$, $p_{labeled} = 0.25$

Class 1	Class 2	Features	Accuracy
butterflies	fish	image	0.91±0.04
butterflies	fish	SDF	**0.93±0.04**
butterflies	heads	image	**1.00±0.01**
butterflies	heads	SDF	**1.00±0.01**
butterflies	dogs	image	**0.97±0.02**
butterflies	dogs	SDF	**0.97±0.02**
fish	heads	image	**0.97±0.01**
fish	heads	SDF	0.96±0.02
fish	dogs	image	**0.95±0.02**
fish	dogs	SDF	0.94±0.03
heads	dogs	image	**0.97±0.01**
heads	dogs	SDF	0.97±0.02

Table 3: The table above shows the accuracy of kNN for image features and SDF features. Neither feature choice is clearly better than the other.

$p_{flip} = 0.2$, $p_{labeled} = 0.05$

Class 1	Class 2	Method	Accuracy
butterflies	fish	kNN	0.69±0.12
butterflies	fish	MR	**0.86±0.10**
butterflies	heads	kNN	0.97±0.01
butterflies	heads	MR	**0.99±0.00**
butterflies	dogs	kNN	0.73±0.16
butterflies	dogs	MR	**0.91±0.04**
fish	heads	kNN	0.73±0.40
fish	heads	MR	**0.98±0.01**
fish	dogs	kNN	0.78±0.23
fish	dogs	MR	**0.89±0.03**
heads	dogs	kNN	0.75±0.22
heads	dogs	MR	**0.93±0.03**

$p_{flip} = 0$, $p_{labeled} = 0.25$

Class 1	Class 2	Method	Accuracy
butterflies	fish	kNN	0.93±0.04
butterflies	fish	MR	**0.94±0.02**
butterflies	heads	kNN	1.00±0.01
butterflies	heads	MR	**1.00±0.00**
butterflies	dogs	kNN	0.97±0.02
butterflies	dogs	MR	**0.98±0.00**
fish	heads	kNN	0.97±0.01
fish	heads	MR	**0.99±0.00**
fish	dogs	kNN	0.95±0.02
fish	dogs	MR	**0.97±0.01**
heads	dogs	kNN	0.97±0.01
heads	dogs	MR	**0.99±0.01**

Table 4: A comparison of the best-performing versions of k-Nearest-Neighbors and Manifold Regularization. Manifold Regularization clearly outperforms kNN.

and RLS become comparable. On real world data sets, where near perfect accuracy is more difficult to achieve, we expect to see a greater performance difference.

6.2. Manifold Regularization vs. kNN

While LapRLS exploits exploits the geometry of the training data by making use ofs the unlabeled data, kNN does not use unlabeled data at all. Under harsh conditions, MR greatly outperforms kNN for all datasets except "butterflies" vs. "heads" where it only outperforms kNN by a small margin. Under mild conditions, however, the performance of kNN is comparable to that of MR, and it might not be worth implementing MR methods. However, the datasets used are artificial and the performance differences on real world data might be more dramatic.

6.3. SDF Features vs. Image Features

SDFs features outperform image features, sometimes dramatically so under harsh conditions such as sparse or noisy labels. We believe this to be because SDFs are smoother than raw images and better obey the manifold assumption. To get a sense for this, consider binary images of handwritten 2s. Two 2s might be very similar in shape, yet have almost no overlap because they are so narrow in most places. The SDFs of those shapes, however, will overlap significantly because information about the shape boundary is propagated across all values of the feature vector. Thus, shapes that would be considered close are in fact close in the feature space, leading to a more well-behaved manifold.

References

Mikhail Belkin, Partha Niyogi, Vikas Sindhwani, and Peter Bartlett. Manifold regularization: A geometric framework for learning from examples. Technical report, Journal of Machine Learning Research, 2004.

Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. On manifold regularization, 2005.

Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *J. Mach. Learn. Res.*, 7: 2399–2434, December 2006. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1248547.1248632.

Tony Chan and Wei Zhu. Level set based shape prior segmentation. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2 - Volume 02*, CVPR '05, pages 1164–1170, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2372-2. doi: 10.1109/CVPR.2005.212. URL http://dx.doi.org/10.1109/CVPR.2005.212.

M. Gong, L. Jiao, L. Bo, L. Wang, and X. Zhang. Image texture classification using a manifold-distance-based evolutionary clustering method. *Optical Engineering*, 47(7): 077201, 2008. doi: 10.1117/1.2955785.

S.S. Haykin. *New directions in statistical signal processing: from systems to brain*. Neural information processing series. MIT Press, 2007. ISBN 9780262083485. URL http://books.google.com/books?id=IlZRAAAAMAAJ.

Bryan C. Russell, Antonio Torralba, Kevin P. Murphy, and William T. Freeman. Labelme: A database and web-based tool for image annotation. *Int. J. Comput. Vision*, 77(1-3):157–173, May 2008. ISSN 0920-5691. doi: 10.1007/s11263-007-0090-8. URL http://dx.doi.org/10.1007/s11263-007-0090-8.

Andy Tsai, Anthony J. Yezzi, William M. Wells III, Clare M. Tempany, Dewey Tucker, Ayres C. Fan, W. Eric L. Grimson, and Alan S. Willsky. A shape-based approach to the segmentation of medical imagery using level sets. *IEEE Trans. Med. Imaging*, 22(2): 137–154, 2003.

Andy Tsai, William M. Wells III, Simon K. Warfield, and Alan S. Willsky. An em algorithm for shape classification based on level sets. *Medical Image Analysis*, 9(5):491–502, 2005.

Oncel Tuzel, Fatih Porikli, and Peter Meer. Pedestrian detection via classification on riemannian manifolds. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(10):1713–1727, October 2008. ISSN 0162-8828. doi: 10.1109/TPAMI.2008.75. URL http://dx.doi.org/10.1109/TPAMI.2008.75.

Rong Zhu, Min Yao, and Yiming Liu. Image classification approach based on manifold learning in web image mining. In Ronghuai Huang, Qiang Yang, Jian Pei, Joo Gama, Xiaofeng Meng, and Xue Li, editors, *Advanced Data Mining and Applications*, volume 5678 of *Lecture Notes in Computer Science*, pages 780–787. Springer Berlin / Heidelberg, 2009. URL http://dx.doi.org/10.1007/978-3-642-03348-3_83. 10.1007/978-3-642-03348-3$_8$3.