Simultaneous Kernel Learning and Label Imputation for Pattern Classification with Partially Labeled Data

Minyoung Kim

Department of Electronics & IT Media Engineering, Seoul National University of Science & Technology, Seoul, Korea

ljfis

Abstract

The kernel function plays a central role in modern pattern classification for its ability to capture the inherent affinity structure of the underlying data manifold. While the kernel function can be chosen by human experts with domain knowledge, it is often more principled and promising to learn it directly from data. This idea of kernel learning has been studied considerably in machine learning and pattern recognition. However, most kernel learning algorithms assume fully supervised setups requiring expensive class label annotation for the training data. In this paper we consider kernel learning in the semi-supervised setup where only a fraction of data points need to be labeled. We propose two approaches: the first extends the idea of label propagation along the data similarity graph, in which we simultaneously learn the kernel and impute the labels of the unlabeled data. The second aims to minimize the dual loss in the support vector machines (SVM) classifier learning with respect to the kernel parameters and the missing labels. We provide reasonable and effective approximate solution methods for these optimization problems. These approaches exploit both labeled and unlabeled data in kernel learning, where we empirically demonstrate the effectiveness on several benchmark datasets with partially labeled learning setups.

Keywords: Kernel learning, Semi-supervised learning, Pattern classification, Optimization

1. Introduction

Pattern classification is a central topic in recent machine learning and pattern recognition, where the goal is to learn an accurate class prediction function given a finite amount of labeled training examples. The prediction function not only explains the training data faithfully, but also needs to be generalizable to unseen data points. The kernel machines have been believed to be one of the most successful pattern recognition approaches for their ability to incorporate potentially nonlinear affinity structures in data through the kernel functions. The support vector machines (SVM) [1] and the Gaussian processes [2] are two most popular algorithms/frameworks based on the kernel functions.

How to choose the kernel function is crucial to success in classification. If human experts or domain knowledge are available, one can exploit them to derive a good kernel function. However, resorting to human efforts is costly, and for certain applications the domain knowledge is inherently unavailable. The kernel learning is a principled solution in such situations

Received: Nov. 23, 2016
 Revised : Dec. 12, 2016
 Accepted: Dec. 14, 2016

Correspondence to: Minyoung Kim
 (mikim@seoultech.ac.kr)
 ©The Korean Institute of Intelligent Systems

©This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (<http://creativecommons.org/licenses/by-nc/3.0/>) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

where it directly learns the kernel function from data. The kernel learning has received significant attention in machine learning and pattern recognition communities, and there is a line of important previous work that has shown the effectiveness of the kernel learning in theory and practice [3–11].

However, most kernel learning methods typically aim to minimize the label prediction loss for a classifier built on the kernel function. In other words, most kernel learning algorithms assume supervised learning setups in which the data points are fully annotated with class labels. Labeling a large number of data points is expensive requiring human efforts, and it is even prohibited when the data size increases drastically. In this paper we deal with kernel learning in partially labeled data setups where only a small portion of data points require to be labeled, and the rest unlabeled data are also exploited to learn a more accurate kernel. That is, we deal with the kernel learning under semi-supervised learning scenarios.

Specifically we propose two approaches to tackle the problem. The first approach extends the idea of the manifold regularization [12] that aims to impute the missing labels using the label propagation along the data similarity manifold formed by the underlying kernel function. Whilst the kernel is assumed given and fixed in this previous work, we extend it for kernel learning formulation. The second method aims to minimize the dual optimum of the SVM classifier where we pose a mini-max optimization that effectively finds the best kernel that yields the smallest SVM hinge loss. In both formulations, we simultaneously optimize the kernel parameters and the missing labels as optimization variables. Although these proposed optimization problems are difficult to solve, we introduce efficient approximate solution methods that can find viable solutions.

The rest of the paper is organized as follows. After briefly reviewing some background on kernel machines and kernel learning, we describe our kernel learning approaches in semi-supervised data setups in Section 3. Empirical evaluations on some benchmark datasets in Section 4 demonstrate the effectiveness of the proposed algorithms, significantly outperforming existing methods that exploit the labeled data only.

2. Background on Kernel Learning

We begin with a brief review of kernel machines for pattern classification problems in machine learning. In the standard binary fully-supervised learning setup, one is given labeled training data points $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, each of which is an iid sample from an unknown distribution $P(x, y)$. Here $x \in \mathbb{R}^p$ is

the p -variate input vector and $y \in \{+1, -1\}$ is the binary class label of x . The goal of pattern classification is to learn a function (called *classifier*) $g : \mathbb{R}^p \rightarrow \{+1, -1\}$ that can accurately predict the label of an unseen data point x as $y = g(x)$.

A typical approach is to consider a linear function family $f(x) = w^\top x$ parameterized by $w \in \mathbb{R}^p$, where the sign of $f(x)$ indicates the class label (+1 or -1). Within the family, learning a classifier f can be typically formulated as empirical loss minimization [1], namely

$$\min_w \sum_{i=1}^n I(y_i \neq \text{sign}(f(x_i))) + \frac{C}{2} \|w\|^2. \quad (1)$$

In (1), $I(\cdot)$ is the indicator function (i.e., 1/0 if the argument is true/false). Thus the first term in the objective measures the training error of the classifier f . The second term is the model regularizer penalizing non-smooth classifiers, required to avoid overfitting and yield good generalization performance [13]. The trade-off parameter $C \geq 0$ balances two objectives.

It can be shown that (e.g., the representer theorem [14]) the optimal w in (1) is a linear combination of the training inputs, i.e., $w = \sum_{i=1}^n \alpha_i x_i$ for some $\alpha_i \in \mathbb{R}$. In other words, the optimal classifier can be always written as: $f(x) = \sum_i \alpha_i x_i^\top x$, namely the linear combination of the inner products between input vectors. The idea can be generalized to a more general non-linear classifier family by replacing x with some nonlinear feature map: $x \rightarrow \phi(x)$. Similarly as linear cases, to represent an optimal classifier, it is sufficient to define the inner product function between feature vectors, which is called the kernel function $k(x, x') = \phi(x)^\top \phi(x')$. That is, by defining the kernel function properly, one needs not deal with an explicit form of the nonlinear feature mapping $\phi(\cdot)$. Specifically, the optimal classifier can be expressed by kernel values only: $f(x) = \sum_{i=1}^n \alpha_i k(x_i, x)$.

The kernel function, as it is the inner product in the feature space, naturally captures similarity between input data points. In kernel machines one can represent arbitrary non-linear affinity structures in data manifold through the kernel functions, without the effort of learning the complex input feature mapping explicitly. This is the major benefit of the kernel machines including SVM and Gaussian processes. For computational simplicity, one typically opts for certain parametric kernel forms. The radial basis function (RBF) kernel, defined as $k(x, x') = \exp(-0.5\|x - x'\|^2/\sigma^2)$ for some scale parameter σ , is one of the most popular kernels used in pattern classification.

Whereas the kernel function or the data affinity structure can be selected from human experts, it is expensive, and often times the domain information is not available at all. A more reasonable treatment is to form a parametric family of the kernel functions, and learn the kernel parameters from data. This idea has been studied considerably in machine learning, referred to as kernel learning [3–11]. Note that the popular model selection approach in kernel machines, namely selecting kernel hyperparameters by cross validation, can be seen as a kernel learning, but the latter typically deals with more sophisticated approaches to kernel function learning. We briefly review two most successful kernel learning methods. Formally we have a parametric family of kernel functions, and we denote a kernel in the family by k_β where β is the kernel parameters. Both approaches assume that one is given fully labeled data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$.

The first approach is the so-called kernel target alignment (KTA) [3]. The KTA aims to maximize the alignment score between the kernel matrix (the similarity matrix on the input vectors) and the label affinity matrix (the similarity matrix on the labels). Here the alignment score between two matrices is defined as the cosine angle between their vector representations. More formally, denoting the $(n \times n)$ kernel matrix over data $\mathcal{D}$ by K (i.e., $K_{ij} = k(x_i, x_j)$), and the class label affinity matrix by O (i.e., $O_{ij} = y_i y_j$), the KTA solves:

$$\max_{\beta} \frac{\langle K, O \rangle_F}{\sqrt{\langle K, K \rangle_F \langle O, O \rangle_F}} = \frac{\langle K, O \rangle_F}{n \sqrt{\langle K, K \rangle_F}}. \quad (2)$$

Here $\langle A, B \rangle_F = \text{tr}(AB^\top)$, and the equality follows directly from $\langle O, O \rangle_F = n^2$. For certain parametric kernel families (e.g., convex combinations of the fixed base kernels in the next section), the optimization (2) reduces to a convex program, which can be solved computationally very efficiently.

The second approach directly minimizes the loss of the kernel classifier based on the underlying kernel. As described above, the optimal classifier has the form of the weighted kernel sum, and its regularized classification loss can be expressed as:

$$R(\beta) = \min_{\alpha} \sum_{i=1}^n l(y_i, f(x_i)) + \frac{C}{2} \|f(\cdot)\|^2$$

$$\text{where } f(x) = \sum_{i=1}^n \alpha_i k(x, x_i). \quad (3)$$

In (3), one can choose the loss function $l(\cdot)$ for any differentiable surrogate of the 0/1 loss for computational tractability. Also, by the kernel trick, the norm of the nonlinear function

$\|f(\cdot)\|^2$ in the objective can be replaced by $\sum_{i,j} \alpha_i \alpha_j k(x_i, x_j)$. Then the kernel learning is done by minimizing $R(\beta)$ with respect to β . In [4], where the SVM's hinge loss and linearly parameterized kernel family are adopted, it is shown that the kernel learning problem can be turned into a semi-definite program (SDP), an instance of convex optimization.

3. Semi-Supervised Kernel Learning

While the previous kernel learning approaches have shown great success in various application areas, the main drawback is that they assume fully-supervised setups. Accurately and manually labeling data points is costly, and in turn it hinders incorporating a large number of training data. In this section we tackle the kernel learning problem in a semi-supervised setup where only a small fraction of the training data points are labeled. Formally the training data is partitioned into labeled and unlabeled sets, $\mathcal{D} = \mathcal{D}_L \cup \mathcal{D}_U$ where $\mathcal{D}_L = \{(x_i, y_i)\}_{i=1}^l$ is the labeled set of l points, and $\mathcal{D}_U = \{(x_i)\}_{i=l+1}^n$ is the unlabeled set of $(n - l)$ points. We typically have $l \ll n$ meaning that only a small fraction of data points are labeled.

While the kernel function can be chosen from a specific functional form like the RBF kernel, it is more flexible to have a combined form of different types of kernel functions. We select and fix M base kernel functions, $k^{(m)} : \mathbb{R}^p \times \mathbb{R}^p \rightarrow \mathbb{R}$ for $m = 1, \dots, M$, and consider the kernel family ($\mathcal{K}$) as a convex hull of the base kernels, namely $\mathcal{K} = \{k \mid k(\cdot, \cdot) = \sum_{m=1}^M \mu_m k^{(m)}(\cdot, \cdot), \mu_m \geq 0, \sum_m \mu_m = 1\}$. That is, μ is the kernel parameter vector we need to learn. We impose the simplex constraints on μ to preserve positive definiteness of the constructed kernel functions with proper scales.

Often we denote the $(n \times n)$ kernel matrix over $\mathcal{D}$ by K (i.e., $K_{ij} = k(x_i, x_j)$), and similarly for base kernel matrices $K^{(m)}$. It is obvious that $K = \sum_{m=1}^M \mu_m K^{(m)}$. Output labels are also denoted as vector forms: $Y_L = [y_1, \dots, y_l]^\top$, $Y_U = [y_{l+1}, \dots, y_n]^\top$, and $Y = [y_1, \dots, y_n]^\top$. To differentiate the ground-truth labels in the training data from the label variables, we use the overline notation ($\bar{Y}_L$) for the former.

In what follows we propose two semi-supervised kernel learning algorithms. In both approaches we formulate optimization problems for the kernel parameters as well as the unknown labels Y_U . That is, the proposed approaches simultaneously learn the kernel and perform the label imputation for unlabeled data.

3.1 Kernel Learning with Label Propagation

The kernel function determines the affinity (manifold) structure of the data clouds, and one can enforce the labels of the labeled data points to propagate to unlabeled points along the manifold. This idea of label propagation has been studied in machine learning, referred to as manifold regularization [12, 15]. Specifically, they minimize an objective function that penalizes the label mismatch between two data points weighted by their kernel value (i.e., similarity), forcing data points closer to each other to attain similar labels, and vice versa. With the ground-truth label constraints for the labeled data points, this has an effect of label propagation from the labeled to the unlabeled set along the affinity manifold.

We extend the idea of manifold regularization to simultaneous kernel learning and label imputation. Note that the manifold regularization is purely semi-supervised label imputation method that assumes the data kernel (affinity structure) is fixed and given. Our kernel learning algorithm based on label propagation can be formulated as follows:

$$\begin{aligned} \min_{\mu, Y} \quad & \frac{1}{2} \sum_{i,j} K_{ij} (y_i - y_j)^2 = Y^\top \mathcal{L} Y \\ \text{s.t.} \quad & K = \sum_{m=1}^M \mu_m K^{(m)}, \quad \mu \geq 0, \quad \sum_m \mu_m = 1, \\ & Y_L = \bar{Y}_L. \end{aligned} \quad (4)$$

The equality in the objective straightforwardly follows by introducing the graph Laplacian $\mathcal{L} = S - K$ where S is the diagonal matrix with row (or column) sums of K in diagonal entries (i.e., $S = \text{diag}(Ke)$ where e is the vector of all ones). Hence, in the above optimization $\mathcal{L}$ is dependent on μ linearly.

As we have both μ and Y as optimization variables, (4) becomes non-convex optimization. Even more, as the label variables Y are constrained to be $\{\pm 1\}$ -valued, it becomes an instance of mixed integer programming which is often more difficult to solve. To remedy this, we first relax the integer-valued constraint by allowing Y to be real-valued in the interval $[-1, 1]$. The final solution Y will then be converted back to the integer values by simple rounding off (i.e., imputing to the closest integer value either $+1$ or -1). For the relaxed problem, we basically perform the projected gradient descent. That is, we repeat two steps until convergence: i) gradient descent $\theta \leftarrow \theta - \eta \nabla J(\theta)$, followed by ii) projection $\theta \leftarrow \Pi(\theta)$. Here θ indicates the optimization variables (either μ or Y), $J(\theta) = Y^\top \mathcal{L} Y$ is the objective function in (4), η is small

positive learning rate, and $\Pi(\theta)$ is the projection operator into the feasible space (for μ we zero out negatives entries followed by normalization to sum up to 1, and for Y we first fix Y_L part as $\bar{Y}_L$ followed by imputing any entries outside the interval $[-1, 1]$ to the closest integer value either $+1$ or -1).

3.2 Kernel Learning with SVM Dual Loss Minimization

As a second approach, we do kernel learning with the objective of minimizing the hinge loss for the SVM classifier built on the kernel. While the main formulation is motivated from the SVM dual optimum minimization in [4], however, unlike their supervised treatment, we deal with semi-supervised data by simultaneously optimizing the labels of the unlabeled data. Specifically, our SVM dual loss based semi-supervised kernel learning can be formulated as:

$$\begin{aligned} \min_{\mu, Y} \quad & \max_{0 \leq \alpha \leq C, \alpha^\top Y = 0} (2\alpha^\top e - \alpha^\top D(Y) K D(Y) \alpha) \\ \text{s.t.} \quad & K = \sum_{m=1}^M \mu_m K^{(m)}, \quad \mu \geq 0, \quad \sum_m \mu_m = 1, \\ & Y_L = \bar{Y}_L, \end{aligned} \quad (5)$$

where e is the all-1 vector, and $D(Y)$ is the diagonal matrix that contains Y as diagonal entries.

In this formulation, presuming that the labels are fully known (i.e., Y is known), the inner maximization part is the SVM dual optimum. This dual optimum coincides with the primal optimal regularized hinge loss due to the strong duality [16] of the SVM's quadratic programming. In the formulation we aim to minimize this dual loss with respect to both kernel parameters μ and the labels Y simultaneously. Given the ground-truth labels for the labeled set, we add the constraints $Y_L = \bar{Y}_L$ as before.

Unlike the semi-definite programming transformation in [4] with fully labeled data, having Y as optimization variables makes the problem hard to solve. We propose an upper-bound minimization technique, where the main idea is to approximate the inner constrained maximization part to its upper bound derived by dropping constraints. Specifically,

$$\begin{aligned} \max_{0 \leq \alpha \leq C, \alpha^\top Y = 0} (2\alpha^\top e - \alpha^\top D(Y) K D(Y) \alpha) \\ \leq \max_{\alpha} (2\alpha^\top e - \alpha^\top D(Y) K D(Y) \alpha) \end{aligned} \quad (6)$$

$$= e^\top (D(Y) K D(Y))^{-1} e \quad (7)$$

$$= e^\top D(Y) K^{-1} D(Y) e \quad (8)$$

$$= Y^\top K^{-1} Y. \quad (9)$$

Table 1. Statistics of the UCI datasets

Dataset	Number of data points	Input dimension
Sonar	208	60
Vote	320	16
Wpbc	180	33
Liver	345	6

In (6) we removed the constraints on α to have an upper bound, and in (7) we have the maximum of the quadratic function in a closed form when $\alpha = (D(Y)KD(Y))^{-1}e$. Also, in (8) we use the fact that $D(Y)^{-1} = D(Y)$.

The objective in (5) is then replaced by (9), and we apply the projected gradient method with $[-1, 1]$ -relaxed Y variables similarly as Section 3.1. In the gradient descent step, the partial derivative of the objective $Y^\top K^{-1}Y$ with respect to μ can be obtained by the following derivation (The partial derivative with respect to Y is simply $2K^{-1}Y$) for each $m = 1, \dots, M$:

$$\frac{\partial Y^\top K^{-1}Y}{\partial \mu_m} = -Y^\top K^{-1} \frac{\partial K}{\partial \mu_m} K^{-1}Y \quad (10)$$

$$= -Y^\top K^{-1} K^{(m)} K^{-1}Y. \quad (11)$$

4. Experiments

In this section we empirically demonstrate the effectiveness of the proposed semi-supervised kernel learning algorithms on several benchmark pattern classification datasets.

We first describe the kernel function hypothesis used throughout the experiments. As mentioned in Section 3 we consider a kernel function family as a convex hull of some fixed base kernels. Specifically the base kernels take the form of weighted combination of the RBF and linear kernels, namely

$$k(x_i, x_j) = \beta_1 \cdot \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right) + \beta_2 \cdot x_i^\top x_j. \quad (12)$$

In (12) we identify a base kernel one by one by selecting the hyperparameters β_1 , β_2 , and σ . We consider a grid on these parameters, more specifically β_1 and β_2 chosen from $\{0, 0.1, 1.0\}$ with the both-0 case discarded, and the RBF scale σ varying uniformly over the 10 equally spaced grid (on the logarithmic scale) of the interval $[10^{-1}, 10^2]$. This results in $M = (3 \times 3 - 1) \times 10 = 80$ base kernels. Thus the convex combination weight parameters μ is 80-dimensional.

We test the kernel learning algorithms on four binary classifi-

cation datasets from the UCI machine learning repository [17]: Sonar, Vote, Wpbc, and Liver datasets. The descriptions of the datasets can be found in <http://archive.ics.uci.edu/ml/> and references therein. The input dimensions and the numbers of data points are listed in Table 1. For each dataset, we randomly partition data points into 60%/40% training/test sets. The training set is further split into labeled and unlabeled sets randomly. We consider three cases for labeled/unlabeled partitions: the labeled set takes about 10%, 30% and 50% of the training set. This whole procedure is repeated for 10 times, and we report the average test errors.

We list below the competing kernel learning approaches with abbreviations.

- **KTA**: The kernel target alignment [3]. Specifically we solve (2) by transforming it into quadratically constrained quadratic programming. Note that it is a supervised method, and the learning is done with the labeled training set only.
- **SLM**: The SVM loss minimization approach suggested in [4, 5]. We plug in the hinge loss in (3), and solve it using the off-the-shelf SDP solvers. Similar to KTA, the algorithm requires fully labeled data, and we make use of the labeled training set only.
- **SSKL-LP**: Our semi-supervised kernel learning algorithm based on label propagation (Section 3.1).
- **SSKL-SDM**: Our second approach based on SVM dual loss minimization (Section 3.2).

To measure the performance of the kernel learning algorithms, once the kernel is learned, we train the SVM classifier for the labeled training data only with the learned kernel, and evaluate the test errors. We report the test errors averaged over 10 random trials in Table 2. As shown, the existing supervised kernel learning algorithms are inferior to ours due to their ignorance of the large number of unlabeled data points. On the other hand, both of our approaches perform equally better on all datasets and all labeled set proportions consistently. This signifies the importance of exploiting the unlabeled data in conjunction with a few labeled samples in kernel learning. We attain this by label propagation along the kernel affinity manifold in SSKL-LP, and simultaneous label imputation and kernel learning within the SVM dual loss minimization framework.

Table 2. Test errors (%) on the UCI datasets with three different labeled training set proportions

Dataset	Method	Labeled set proportions		
		10%	30%	50%
Sonar	KTA	48.43 $\pm$ 4.42	43.95 $\pm$ 3.23	36.67 $\pm$ 3.94
	SLM	47.14 $\pm$ 3.83	43.10 $\pm$ 3.45	38.81 $\pm$ 3.82
	SSKL-LP	43.05 $\pm$ 3.79	40.24 $\pm$ 3.80	33.10 $\pm$ 3.05
	SSKL-SDM	42.14 $\pm$ 4.22	40.00 $\pm$ 3.81	30.48 $\pm$ 3.73
Vote	KTA	25.86 $\pm$ 3.99	22.66 $\pm$ 3.50	13.57 $\pm$ 4.55
	SLM	23.66 $\pm$ 3.21	20.76 $\pm$ 4.62	13.74 $\pm$ 3.46
	SSKL-LP	19.86 $\pm$ 4.62	16.57 $\pm$ 3.55	11.89 $\pm$ 2.78
	SSKL-SDM	18.29 $\pm$ 3.90	15.29 $\pm$ 3.69	10.97 $\pm$ 3.53
Wpbc	KTA	35.43 $\pm$ 4.60	31.87 $\pm$ 3.97	29.05 $\pm$ 3.13
	SLM	34.39 $\pm$ 3.94	31.45 $\pm$ 4.60	29.87 $\pm$ 4.79
	SSKL-LP	31.82 $\pm$ 3.42	28.43 $\pm$ 3.57	26.06 $\pm$ 3.65
	SSKL-SDM	31.39 $\pm$ 3.65	27.56 $\pm$ 3.13	25.82 $\pm$ 3.42
Liver	KTA	48.84 $\pm$ 4.38	46.80 $\pm$ 3.32	40.87 $\pm$ 3.59
	SLM	47.25 $\pm$ 3.67	45.30 $\pm$ 3.90	40.75 $\pm$ 3.73
	SSKL-LP	42.03 $\pm$ 3.72	40.29 $\pm$ 4.72	38.87 $\pm$ 3.92
	SSKL-SDM	42.88 $\pm$ 3.32	41.30 $\pm$ 3.67	38.72 $\pm$ 3.59

5. Conclusion

In this paper we have proposed novel semi-supervised kernel learning algorithms that exploit a few labeled examples in conjunction with a large amount of unlabeled data points. One of our approaches seeks for the most plausible missing label imputation as well as the kernel similarity structure along which the labels can be propagated with minimal cost. In our second approach, the SVM hinge loss is minimized simultaneously over the kernel function and the missing labels. These approaches are promising in that they exploit both labeled and unlabeled data in principled ways, and indeed the effectiveness is empirically demonstrated on several benchmark datasets with partially labeled learning setups. While we have suggested several approximate solution methods for the optimization problems to deal with difficult mixed integer programming, and they turn out to yield viable solutions, it remains as future work to devise more efficient and robust solution algorithms.

Conflict of Interest

No potential conflict of interest relevant to this article was reported.

Acknowledgements

This study was supported by Seoul National University of Science & Technology.

References

- [1] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY: Springer, 1995.
- [2] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA: MIT Press, 2006.
- [3] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. Kandola, "On kernel-target alignment," *Advances in Neural Information Processing Systems*, vol. 14, 2001.
- [4] G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan, "Learning the kernel matrix with semidefinite programming," *Journal of Machine Learning Research*, vol. 5, pp. 27-72, 2004.
- [5] A. Rakotomamonjy, F. R. Bach, S. Canu, and Y. Grandvalet, "SimpleMKL," *Journal of Machine Learning Research*, vol. 9, pp. 2491-2521, 2008.

- [6] Z. Xu, R. Jin, I. King, and M. Lyu, "An extended level method for efficient multiple kernel learning," *Advances in Neural Information Processing Systems*, vol. 21, pp. 1825-1832, 2008.
- [7] M. Varma and B. R. Babu, "More generality in efficient multiple kernel learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*, Montreal, Canada, 2009. <http://dx.doi.org/10.1145/1553374.1553510>
- [8] F. Bach, "Exploring large feature spaces with hierarchical multiple kernel learning," *Advances in Neural Information Processing Systems*, vol. 21, pp. 105-112, 2008.
- [9] P. Gehler and S. Nowozin, "On feature combination for multiclass object classification," in *Proceedings of 2009 IEEE 12th International Conference on Computer Vision*, Kyoto, Japan, 2009.
- [10] Z. Yang, A. J. Smola, L. Song, and A. G. Wilson, "A la carte: learning fast kernels," in *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*, San Diego, CA, 2015.
- [11] A. G. Wilson, Z. Hu, R. Salakhutdinov, and E. P. Xing, "Deep kernel learning," in *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, Cadiz, Spain, 2016.
- [12] M. Belkin, P. Niyogi, and V. Sindhwani, "On manifold regularization," in *Proceedings of the 10th International Conference on Artificial Intelligence and Statistics*, Barbados, 2005.
- [13] B. Schoolkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA: MIT Press, 2002.
- [14] B. Schoolkopf, R. Herbrich, and A. Smola, "A Generalized representer theorem," *Computational Learning Theory*, vol. 2111, pp. 416-426, 2001. http://dx.doi.org/10.1007/3-540-44581-1_27
- [15] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using Gaussian fields and harmonic functions," in *Proceedings of the 20th International Conference on Machine Learning*, Washington, DC, 2003.
- [16] R. T. Rockafellar, *Convex Analysis*. Princeton: Princeton University Press, 1997.
- [17] A. Asuncion and D. J. Newman, "UCI machine learning repository," Available <http://www.ics.uci.edu/~mllearn/MLRepository.html>

Minyoung Kim received his BS and MS degrees both in Computer Science and Engineering from Seoul National University, South Korea. He earned a PhD degree in Computer Science from Rutgers University in 2008. From 2009 to 2010 he was a postdoctoral researcher at the Robotics Institute of Carnegie Mellon University. He is currently an associate professor in Department of Electronics and IT Media Engineering at Seoul National University of Science and Technology in Korea. His primary research interest is machine learning and computer vision. His research focus includes graphical models, motion estimation/tracking, discriminative models/learning, kernel methods, and dimensionality reduction. E-mail: mikim@seoultech.ac.kr