6-2010

Weakly-Supervised Hashing in Kernel Space

Yadong MU
National University of Singapore

Jialie SHEN
Singapore Management University, jlshen@smu.edu.sg

Shuicheng YAN
National University of Singapore

DOI: https://doi.org/10.1109/CVPR.2010.5540024

Published in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010:
San Francisco, California, USA, 13-18 June 2010, pp. 3344-3351.
http://doi.org/10.1109/CVPR.2010.5540024

Weakly-Supervised Hashing in Kernel Space

Yadong Mu[1], Jialie Shen[2], Shuicheng Yan[1]
[1]National University of Singapore, {elemy, eleyans}@nus.edu.sg
[2]Singapore Management University, jlshen@smu.edu.sg

Abstract

The explosive growth of the vision data motivates the recent studies on efficient data indexing methods such as locality-sensitive hashing (LSH). Most existing approaches perform hashing in an unsupervised way. In this paper we move one step forward and propose a supervised hashing method, i.e., the LAbel-regularized Max-margin Partition (LAMP) algorithm. The proposed method generates hash functions in weakly-supervised setting, where a small portion of sample pairs are manually labeled to be "similar" or "dissimilar". We formulate the task as a Constrained Convex-Concave Procedure (CCCP), which can be relaxed into a series of convex sub-problems solvable with efficient Quadratic-Program (QP).

The proposed hashing method possesses other characteristics including: 1) most existing LSH approaches rely on linear feature representation. Unfortunately, kernel tricks are often more natural to gauge the similarity between visual objects in vision research, which corresponds to probably infinite-dimensional Hilbert spaces. The proposed LAMP has a natural support for kernel-based feature representation. 2) traditional hashing methods assume uniform data distributions. Typically, the collision probability of two samples in hash buckets is only determined by pairwise similarity, unrelated to contextual data distribution. In contrast, we provide such a collision bound which is beyond pairwise data interaction based on Markov random fields theory.

Extensive empirical evaluations are conducted on five widely-used benchmarks. It takes only several seconds to generate a new hashing function, and the adopted random supporting-vector scheme enables the LAMP algorithm scalable to large-scale problems. Experimental results well validate the superiorities of the LAMP algorithm over the state-of-the-art kernel-based hashing methods.

1. Introduction

Recent rapid growth of visual data brought by Internet has erose great interest in techniques for efficient data organization and data access for various vision applications. The key research problems include similarity metric learning between pairwise data, and retrieval of nearest neighbors given any query datum. For the former, numerous algorithms have been proposed, especially the kernel-induced metric [8]. While for the latter problem, linear scan is an intuitive and common method, yet computationally forbidden for large-scale datasets, which spurs the development of various *approximate-nearest-neighbor* (ANN) algorithms. A bunch of ANN algorithms were proposed based on the concept of *locality-sensitive hashing* (LSH) [7]. Given a similarity metric $\varkappa(\cdot,\cdot)$ in the feature space, the LSH algorithms typically guarantee the probability for any two samples x_i and x_j falling into the same bucket to be the quantity $\varkappa(x_i, x_j)$, known as the "locality sensitive" property.

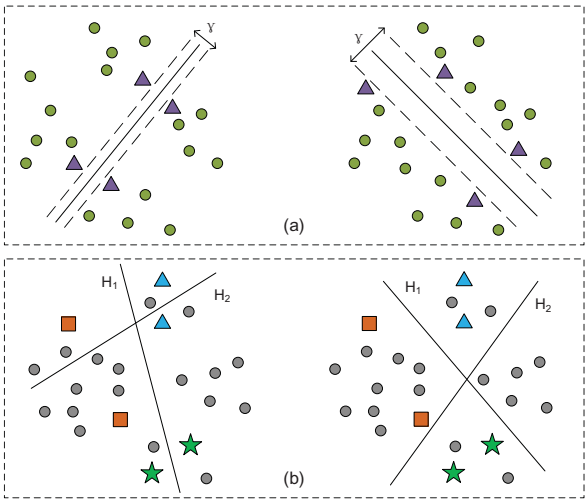

Figure 1. Illustration of the motivations for *maximum margin partition* and *side information regularization* in hashing. Figure (a) shows two hash functions which result in different margin γ on the same distribution. While figure (b) illustrates how side information (in this example, three sample pairs are manually labeled to be "similar", denoted by triangle, stars, and squared boxes respectively) can guide more reasonable hashing scheme. In both cases, results on the right figures are more reasonable.

Despite of the notable success from LSH related algo-

rithms [12, 13], several important issues are seldom tackled in related literature, which are however very common in research topics like object recognition and image annotation:

1. Many vision datasets are constructed accompanied with rich side information, such as the category labels of Caltech-101 images and the tags attached to the Flickr or YouTube data. Traditional hashing methods only guarantee a high collision probability for samples that are near in the feature space. However, due to the semantic gap between low-level features and semantics, feature proximity is sometimes inconsistent with semantic similarity. A straightforward solution is to utilize this side information as regularization term during hash table construction.

2. Kernel tricks are very popular in computer vision research owing to its great potentials in handling nonlinearly separated data. For example, in bag-of-features (BOF) model, images are represented as collections of orderless feature points, and thus the kernel function (e.g., intersection kernel) is a more natural choice for data similarity measure. It arises the challenge for data indexing: how to construct hash table in kernel spaces?

The above two issues motivate our work in this paper. We propose a label-regularized maximum-margin partition (LAMP) method (see Figure 1), to enhance hashing quality by using kernel-based similarity and additional small number of pairwise constraints as side information.

2. Related Works

For similarity search problem, unlike classic KD-Tree [3], LSH is effective even in high-dimensional feature spaces. One popular method in LSH is to generate a random vector h from a particular probabilistic distribution, e.g., p-stable distribution [5] for ℓ^p-metric space, and bucket data according to the sign of the projected value on h. See [1] for a brief survey about LSH-related algorithms. In this work we are particularly interested in kernel-based hashing, which has wide applications in computer vision. The state-of-the-art and possibly the only work was proposed by Kulis et al. in [9], where random hyperplane-based hash functions are generated to build buckets, based on *Central Limit Theorem* (CLT) in kernel-induced Hilbert space. Recall that traditional LSH requires random vectors generated from a particular Gaussian distribution, e.g., with zero-mean and identity covariance matrix, which is unfortunately infeasible in kernel space. The work in [9] performed kernel-based hashing based on a fact revealed by CLT, i.e., for sufficiently many *independent identical distribution* (i.i.d.) samples, their summation $\widetilde{z}_t = \frac{1}{t}\sum_{i=1}^{t} x_i$ is distributed according to a multi-variate Gaussian, and the desired hashing function can be obtained by a whitening transform. Although this assumption only holds under very mild conditions, the proposed kernelized LSH gains success in several vision benchmarks.

To our best knowledge, rare prior work was devoted to hashing with side information. Typically metric learning is performed from these side information beforehand [8], either monolithic, group, or per-category based [2], and traditional hashing is then called afterwards. However, the performance of metric learning heavily depends on the quality of side information, and is not scalable for large-scale vision datasets on the order of millions or billions.

3. The LAMP Algorithm

3.1. Basic Idea

Figure 1 illustrates the main idea of our proposed Label-regularized max-margin partition (LAMP) algorithm. We have the following three observations:

1. Side information, such as "similar" or "dissimilar" constraints, or label (tag) information, provides useful cues for more reasonable hashing results, as seen from the example provided in Figure 1.

2. Larger margin between hash-induced partitions usually indicates better generalization ability to out-of-sample data. Denote the resulting margin as γ. All query data falling within $\frac{1}{2}\gamma$-neighborhood can be correctly judged, thus larger γ potentially implies lower error rate in similarity search.

3. Although Figure 1 illustrates the case in linear feature space, however, image kernels are more natural choices for similarity measure in various vision applications. A hashing algorithm supporting kernel-based representation is especially useful.

3.2. Notations

Suppose we are given a collection of data points $\mathcal{X} = \{x_i\}_{i=1\ldots n}$ where x_i lies in a linear vector space (maybe of infinite dimensionality in kernel space). A hash function can be equivalently regarded as a partition function which divides the original data space into two parts. Denote the hash vector as ω, the real-valued response of datum x_i about ω as $f(x_i)$, and the final partition label as y_i. Based on a margin-oriented criterion, the binary partition task can be formulated as below:

$$\mathcal{J}_\omega = \Omega(\|\omega\|_{\mathcal{H}}) + \sum_i \ell\left(-y_i f(x_i)\right). \tag{1}$$

where Ω and $\ell(\cdot)$ denote the regularization function about functional norm $\|\omega\|_{\mathcal{H}}$ in Hilbert spaces (monotonically increasing), and loss function of "margin" $yf(x)$ (typically

convex) respectively. The most popular definitions are ℓ^2-norm $\Omega(\|w\|_{\mathcal{H}}) = \frac{1}{2}\|w\|^2$ and hinge loss $\ell(x_i) = (1 - y_i f(x_i))_+$, where the subscript $+$ indicates a cutoff at zero to guarantee non-negativity. When handling non-linear data distributions, the so-called "kernel trick" is utilized. Each sample is transformed into an implicit Hilbert space (probably infinite-dimensional) via a mapping function $\phi(\cdot)$. Following the representer theorem [10], ω can be expressed as a linear combination of the mapped data vectors, i.e., $\omega = \sum_{i=1}^{n} \alpha_i \phi(x_i)$, where α_i represents the unknown parameter.

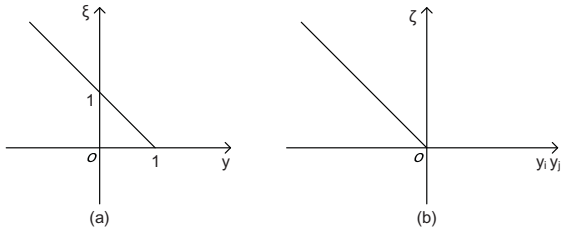

Figure 2. Illustration of the hinge-shaped loss functions $\ell(\cdot)$ in (a) and regularizer $\hbar(\cdot)$ in (b).

In this paper we adopt the term "side information" to denote any user-supplied "similar" or "dissimilar" constraints between two samples. For example, when the data are partially labeled, it is reasonable to expect samples sharing the same label to stay in adjacent hash buckets, toggling an extra penalty when they are projected into different partitions. Hereafter the constraint set for all the side information is denoted as Θ_s. We can then add another regularization term to enforce the consistency between hashing partitions and constraints (for clarity, we assume Θ_s only contains "similar" constraints, and the extension to "dissimilar" case is straightforward):

$$\mathcal{J}_\omega = \Omega(\|\omega\|_{\mathcal{H}}) + \sum_i \ell\left(-y_i f(x_i)\right) + \lambda \sum_{\theta \in \Theta_s} \hbar(\theta). \quad (2)$$

Prior study in [6] adopted Laplacian loss $|f(x_i) - f(y_j)|$ for regularization term $\hbar(\cdot)$. Although retaining convexity, this form, however, triggers unexpected high penalty when two samples stay in the same partition yet have a large distance. To avoid this issue, here we propose an alternative definition of the regularizer $\hbar(\cdot)$ as below:

$$\hbar(\theta_{i,j}) = \begin{cases} 0, & y_i = y_j \\ (-y_i y_j)_+, & y_i \neq y_j \end{cases} \quad (3)$$

It is linear and convex with respect to $y_i y_j$, analogous to the traditional hinge loss in *Supporting Vector Machine* (SVM). See Figure 2 for an intuitive comparison.

Following the traditional notations in max-margin formulation [11], we arrive at the following optimization problem:

$$\begin{aligned} \min_{\omega,b,\xi,\zeta,y} \quad & \frac{1}{2}\|w\|^2 + \frac{\lambda_1}{n}\sum_i \xi_i + \frac{\lambda_2}{n}\sum_{\theta\in\Theta_s} \zeta_{ij} \quad (4) \\ s.t. \quad & y_i(\omega^T x_i + b) + \xi_i \geq 1,\ \xi_i \geq 0,\ \forall\, i, \\ & y_i y_j + \zeta_{ij} \geq 0,\ \zeta_{ij} \geq 0,\ \forall\, (i,j) \in \Theta_s. \end{aligned}$$

3.3. Random SV and Relaxation

For data sets of large size, the problem in (4) is computationally prohibitive. In this work we propose the idea of "random supporting vectors" to reduce the computational burden. Namely, p data are randomly sampled from $\mathcal{X}$ and serve as the supporting vectors (SV). In other words, ω is expressed as below:

$$\begin{aligned} \omega &= \nu_1\phi(x_1) + \nu_2\phi(x_2) + \ldots + \nu_p\phi(x_p) \\ &= [\phi(x_1), \cdots, \phi(x_p)]\,\nu \end{aligned}$$

Moreover, since in LSH-related algorithms y_i is always set to be $\text{sign}(\omega^T x_i + b)$, then $y_i(\omega^T x_i + b)$ is consequently non-negative and can be equivalently expressed as $|\omega^T x_i + b|$. Another crucial relaxation of the formulation is to replace $y_i y_j$ with $(\omega^T x_i + b)(\omega^T x_j + b)$, such that variables y_i's are eliminated in the optimization. Finally, we get the modified formulation:

$$\begin{aligned} \min_{\nu,b,\xi,\zeta} \quad & \frac{1}{2}\nu^T G \nu + \frac{\lambda_1}{n}\sum_i \xi_i + \frac{\lambda_2}{n}\sum_{\theta\in\Theta_s} \zeta_{ij} \quad (5) \\ s.t. \quad & |\nu^T k_i + b| + \xi_i \geq 1, \quad (6) \\ & (\nu^T k_i + b)(\nu^T k_j + b) + \zeta_{ij} \geq 0, \quad (7) \\ & \xi_i \geq 0,\ \forall\, i, \\ & \zeta_{ij} \geq 0,\ \forall\, (i,j) \in \Theta_s, \end{aligned}$$

where G is a $p \times p$ Gram matrix computed from the p random supporting vectors, and k_i denotes the inner products between the i-th sample and all p supporting vectors.

To optimize Problem (5), however, is difficult since it is non-convex and nonlinear. In the following subsections, we first point out its relationship with *constrained-concave-convex-procedure* (CCCP) (Section 3.4), and then present the relaxed convex sub-problems in each iteration (Section 3.5), together with the efficient cutting-plane based QP solver (Section 3.6). An overview is provided in Algorithm 1.

3.4. Concave-Convex Procedure

Notable acceleration is possible based on the observation that Problem (5) is actually a special case of *constrained-concave-convex-procedure* (CCCP) [17, 15], which targets at optimization problems in the following forms:

$$\begin{aligned} \min_x \quad & f_0(x) - g_0(x) \quad (8) \\ s.t. \quad & f_i(x) - g_i(x) \leq c_i, \quad i = 1, \ldots, m, \end{aligned}$$

for $t = 1 \ldots t_{max}$ **do**
  Relax the CCCP problem into convex sub-problem $\mathcal{J}_t$ using Taylor expansion (Section 3.5), and initialize $(\omega_{t,0}, b_{t,0}) = (\omega_t, b_t)$;
  for $k = 1 \ldots k_{max}$ **do**
    1. Calculate the cutting planes at location $(\omega_{t,k}, b_{t,k})$ and add into the constraint set;
    2. Use Quadratic-Program to solve the reduced cutting-plane problem (Section 3.6) and obtain a new solution $(\omega_{t,k+1}, b_{t,k+1})$;
  end
end

Algorithm 1: The LAMP algorithm

where f_i and g_i are both real-valued convex functions. In other words, both the objective and m constraints are the difference of two convex functions. Assume the Hessians of all g_i are positive semi-definite, in the t-th iteration an upper bound of objective value can be achieved by replacing g_i with its first-order Taylor expansion around current solution x_t, i.e., $\mathcal{T}(g(x_t)) = g(x_t) + \partial_x g(x_t)(x - x_t)$, where $\partial_x g(x_t)$ denotes the first-order derivative of $g(x)$ at x_t. The sub-problem obtained by Taylor expansion is in a simpler convex form and can be solved by off-the-shelf convex solvers. The value of parameter x is then updated to get x_{t+1}. Given an initial value x_0, the solution series $\{x_t\}$ obtained by CCCP is guaranteed to reach a local optimum.

For Problem (5), the objective is convex and the constraint (6) is difference-of-convex ($|\nu^T k_i + b|$ is convex). And constraint (7) proves to be convex after some transforms. For clarity we use the abbreviation $f_i = \nu^T k_i + b$ and $f_j = \nu^T k_j + b$, and then we have,

$$\begin{aligned} f_i f_j + \zeta_{ij} \geq 0 \quad &\Leftrightarrow \quad f_i^2 + f_j^2 + 2 f_i f_j + 2\zeta_{ij} \geq f_i^2 + f_j^2 \\ &\Leftrightarrow \quad \frac{1}{2}(f_i + f_j)^2 + \zeta_{ij} \geq \frac{1}{2} f_i^2 + \frac{1}{2} f_j^2 \end{aligned}$$

After introducing a new notation $\widetilde{\nu} = \begin{pmatrix} \nu \\ b \end{pmatrix}$, the above inequality can be further transformed into the following difference-of-convex form:

$$\begin{aligned} \frac{1}{2}\widetilde{\nu}^T (M_i + M_j)\widetilde{\nu} - \frac{1}{2}\widetilde{\nu}^T M_{ij}\widetilde{\nu} - \zeta_{ij} \leq 0 \\ M_i \succeq 0, \;\; M_j \succeq 0, \;\; M_{ij} \succeq 0 \end{aligned}$$

where the operator $\succeq$ denotes *positive semi-definite*. Matrices M_i, M_j and M_{ij} can be calculated from k_i and k_j. For example, it is easy to verify that

$$M_{ij} = \begin{bmatrix} k_i k_j^T & \frac{1}{2}(k_i + k_j) \\ \frac{1}{2}(k_i + k_j)^T & 1 \end{bmatrix},$$

thus Problem (5) is a CCCP problem, and can be solved by general CCCP solutions.

3.5. Taylor Expansion at t-th Iteration

Before proceeding, this subsection elaborates on the detailed convex sub-problem at the t-th CCCP iteration. Firstly, note that $|\nu' k_i + b|$ is non-smooth, whose derivative possibly doesn't exist at some locations. In this situation we replace gradient with the subgradient [4] alternatively:

$$\begin{aligned} \partial_\nu |\nu^T k_i + b| &= k_i \cdot \text{sign}(\nu_t^T k_i + b_t), \\ \partial_b |\nu^T k_i + b| &= \text{sign}(\nu_t^T k_i + b_t). \end{aligned}$$

The Taylor approximations for constraints (6) and (7) are then expressed as:

$$|\nu^T k_i + b| \approx (\nu^T k_i + b) \cdot \text{sign}(\nu_t^T k_i + b_t), \quad (9)$$

$$\frac{1}{2}\widetilde{\nu}^T M_{ij}\widetilde{\nu} \approx \widetilde{\nu}_t^T M_{ij}\nu - \frac{1}{2}\widetilde{\nu}_t^T M_{ij}\nu_t. \quad (10)$$

The number of slack variables ξ_i equals to the data number while the number of ζ_{ij} equals to the cardinality of set Θ_s. Here we introduce another two variables to reduce the parameter number, i.e., let $\xi = \sum_i \xi_i$ and $\zeta = \sum_{\theta \in \Theta_s} \zeta_{ij}$. Putting everything together, finally we obtain the optimization problem in the t-th iteration:

$$\mathcal{J}_t = \min_{\nu, b, \xi, \zeta} \frac{1}{2}\nu^T G \nu + \frac{\lambda_1}{n}\xi + \frac{\lambda_2}{n}\zeta, \quad (11)$$

which is subject to the following two convex constraints:

$$\xi \geq \sum_i \Big(1 - (\nu^T k_i + b) \cdot \text{sign}(\nu_t^T k_i + b_t)\Big)_+, \quad (12)$$

$$\zeta \geq \sum_{\theta \in \Theta} \Big(\frac{1}{2}\widetilde{\nu}^T (M_i + M_j)\widetilde{\nu} - \widetilde{\nu}_t^T M_{ij}\nu + \frac{1}{2}\widetilde{\nu}_t^T M_{ij}\nu_t\Big)_+.$$

3.6. Optimization with Cutting Plane

CCCP-based relaxation produces a sequence of sub-problem $\mathcal{J}_t$, $t = 0 \ldots t_{max}$ as in (11). However, solving Problem (11) is still difficult and time-consuming since the variables ξ, ζ and $\widetilde{\nu}$ are highly coupled. A practical way is to accelerate using Cutting-Plane (CP) method [14]. In CP terminology, the original Problem (11) is usually called *master problem*. Both variables ξ and ζ can be regarded to be nonlinear functions of $\widetilde{\nu}$. For example, for ξ, we have,

$$\xi(\widetilde{\nu}) = \sum_i \Big(1 - c_t^i \cdot (\nu^T k_i + b)\Big)_+, \quad (13)$$

where $c_t^i = \text{sign}(\nu_t^T k_i + b_t)$ is fixed throughout the t-th CCCP. Assume CP method takes at most k_{max} iterations to converge for optimizing $\mathcal{J}_t$, and denote the optimum in its k-th iteration as $\widetilde{\nu}_{t,k}$. The basic idea of CP method is to maintain a collection of linear constraints, substituting original nonlinear ones like in (12). This constraint set is initialized empty and expanded immediately after obtaining a new

solution $\widetilde{\nu}_{t,k}$ according to the following rule: since $\xi(\widetilde{\nu})$ is convex and has positive semi-definite Hessian matrix, it can be approximated around $\widetilde{\nu}_{t,k}$ by a linear inequality:

$$\xi(\widetilde{\nu}) \geq \xi(\widetilde{\nu}_{t,k}) + \partial_{t,k}\xi \cdot (\widetilde{\nu} - \widetilde{\nu}_{t,k}),\ \forall\, \widetilde{\nu} \in \mathcal{R}^{p+1}, \quad (14)$$

where $\partial_{t,k}\xi$ denotes any subgradient of $\xi(\widetilde{\nu})$ at $\widetilde{\nu}_{t,k}$. The case of $\zeta(\widetilde{\nu})$ is similar. These linear constraints are called *cutting planes* in CP terminology. The optimization proceeds using efficient Quadratic-Program (QP) solver and terminates until no salient gain when adding more cutting-plane constraints. In other words, denote the sub-problem in the t-th CCCP and the k-th Cutting-Plane as $\mathcal{J}_{t,k}$. The optimum sequence $\mathcal{J}^*_{t,k}$, $k = 0 \ldots k_{max}$ monotonically increases until the convergence to the optimal solution of $\mathcal{J}_t$.

3.7. Practical Issues

A trivially "optimal" solution is to assign all data to the same partition, and the resultant margin will be infinite positive. To prevent such a meaningless solution, a partition-balance constraint is required. A possible solution is to enforce $-l \leq \frac{1}{n}\sum_{i=1}^{n}(\nu' k_i + b) \leq l$, where l is a pre-defined constant (fixed to be 0.1 in our implementation).

Another important strategy is to reduce the correlation between randomly-generated hash functions. We capitalize on the similar idea in [16]. Stacking the hash bits y_i, $i = 1 \ldots n$ of all data into a hash-value vector, we prefer lower squared Pearson coefficients between different hash-value vectors. Moreover, this additional penalty can be seamlessly incorporated into the quadratic term $\frac{1}{2}\nu^T G \nu$.

4. Algorithmic Analysis

4.1. Complexity and Convergence

In our formulation, each sub-problem $\mathcal{J}_{t,k}$ is a convex quadratic program with $p + 2$ variables (p represents the number of random supporting vectors) and at most $2k_{max}$ linear constraints, and can be efficiently solved using off-the-shelf convex QP solvers. In practice we use the built-in QP solver in the MOSEK optimization package.

The value of k_{max} reflects the convergence speed of the cutting-plane method. Assume the cutting-plane procedure halts when satisfying the ϵ-optimality condition, i.e., the difference between objective values of the master problem and reduced cutting-plane problem is below a threshold ϵ. An important observation is that k_{max} is actually upper-bounded by a constant only related to ϵ, λ_1 and λ_2. The conclusion follows from two facts: 1) The objective value of Problem (11) is upper-bounded since $(\nu,\ b) = 0$ is a feasible point in the master problem. 2) The amount by which the solution increases by adding one constraint is lower-bounded by a constant determined by ϵ, λ_1 and λ_2 (see [14] for details). Consequently, the algorithm can only perform a constant number of iterations before termination.

In Figure 3 we show the objective value's evolutionary curves of both the reduced cutting-plane problem and the corresponding *master problem* in (11), which are captured during the experiment on the massive MNIST-Digit benchmark. The curves validate the fact that cutting-plane method provides a lower bound of original master problem with convergence guarantee. In practice it typically converges in fewer than 10 iterations. Refer to [15] for the detailed discussion about the convergence property of outer CCCP loop.

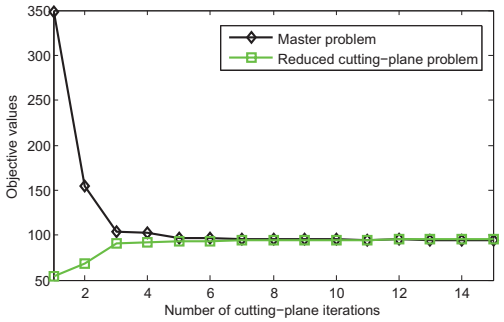

Figure 3. Evolutionary curves of the inner cutting-plane loop.

4.2. Bayesian Collision Bound

In this sub-section we give a bound for two samples' collision probability in the constructed hash buckets from a Bayesian point of view. Recall that the object function to optimize is as follows (shift variable b is ignored for clarity without loss of generality):

$$\mathcal{J}_\omega = \frac{1}{2}\|\omega\|^2 + \frac{\lambda_1}{n}\sum_i (1 - y_i \omega^T x_i)_+ + \frac{\lambda_2}{n}\sum_{\theta \in \Theta}(-y_i y_j)_+.$$

Associating each sample with a random variable, and modeling the data interaction with Markov Random Fields (MRF), the above optimization problem is then equivalent to maximize the following joint probability:

$$P_{\omega,y} \propto \exp\Big(-\frac{1}{2}\|\omega\|^2\Big)\prod_i \exp\Big\{-\frac{\lambda_1}{n}(1 - y_i\omega^T x_i)_+\Big\} \prod_{\theta\in\Theta}\exp\Big\{-\frac{\lambda_2}{n}(-y_i y_j)_+\Big\}, \quad (15)$$

where $\exp(-\frac{1}{2}\|\omega\|^2)$ can be regarded as a priori knowledge about ω, while the other two are uniary and binary potentials respectively. The binary potential reflects the impact of side information. Note that under our proposed *random supporting vectors* scheme, ω is subject to specific parametric-form distribution. Specifically, with in total n training data and p random SVs, denote Π to be a distribution which can generate n-length zero vectors π_i except for p entries being one (the opportunity to get a value one

is identical for every entry). For any $\pi_i \in \Pi$, under a mild assumption that globally optimal solution conditioned on π_i is unique and feasible for some polynomial-time optimization method (cutting-plane method in our implementation, which we suppose can achieve good enough solution that is approximately globally optimal), unique ω^* and y^* are determined under above assumptions by solving the optimization problem:

$$\omega^*,\ y^* = \arg\max_{\omega,y} P_{\omega,y}, \quad \omega \sim \pi_i, \tag{16}$$

The estimations of ω and y consequently determine another binary-valued random variable δ_{ij} which values 1 when $y_i^* = y_j^*$, otherwise 0. Finally we can expressed the collision probability $Pr(y_i = y_j)$ of data $i,\ j$ as an integral of δ_{ij} over distribution Π:

$$Pr(y_i = y_j) = \int \delta_{ij}\, d\pi_i. \tag{17}$$

Note that $Pr(y_i = y_j)$ not only depends on the two data, but also are affected by the contextual data distribution, fundamentally differing from traditional collision bounds like in [5].

5. Experiments

Our evaluations consist of two parts, on benchmarks either with groundtruth labels or not. For each data set, 90% samples are used for hash table construction, and the rest are used for testing. Most databases are of large size (9K $\sim$ 1.7 million), thus we randomly sample n (n = 2000 by default in practice) data from the whole benchmark as working set, which proves to enhance efficacy without much loss of accuracy. On most benchmarks (except for Caltech-101), we adopt the Gaussian kernel $K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{\gamma^2}\right)$, where the scaling factor γ takes three values (0.5σ, σ and 5σ respectively, σ is the standard variation of $\|x_i - x_j\|$). The resultant three kernels are finally combined with uniform weights. For the parameters in our formulation, by default we set $\lambda_1 = 80$ and $\lambda_2 = 10$. For the number of random supporting vectors, we set p = 40 for the first three experiments, and p = 100 for others. It takes roughly $4 \sim 15$ seconds to generate a hash function on our common desktop PC, depending on the maximum iteration before convergence. All experiments repeat 10 times, and the reported averaged results are compared with the state-of-the-art Kernelized Locality-Sensitive Hashing (KLSH) in [9].

5.1. Data Sets with Labels

We adopt three benchmarks with labels as below:
Caltech-101[1] is one of the most popular benchmarks for object recognition, containing 101 distinct categories and one background class. Each class contains tens of examples, forming a median-scale data set ($\approx 9K$ images). We adopt Caltech-101 since many existing algorithms on image kernels [9] use it as a test bed. Specifically, we first extract three kinds of local features, including geometric blur, PCA-SIFT, pyramid histogram of visual words (PHOW). After that, intersection kernels on histograms are calculated.
MNIST-Digit[2] is built for handwritten digits recognition. Among the total 70K examples, there are 7K images for each digit in $0 \sim 9$. In practice, each 28×28 digit image is transformed by matrix-to-vector concatenation and normalized to be unit-length feature.
CIFAR-10[3] is a labeled subset of the 80 million tiny images dataset, containing 60K 32×32 color images in 10 classes (6K images for each class). The dataset is constructed to learn meaningful recognition-related image filters whose responses resemble the behavior of human visual cortex. In the experiment we use the 387-d GIST image feature and Gaussian kernels.

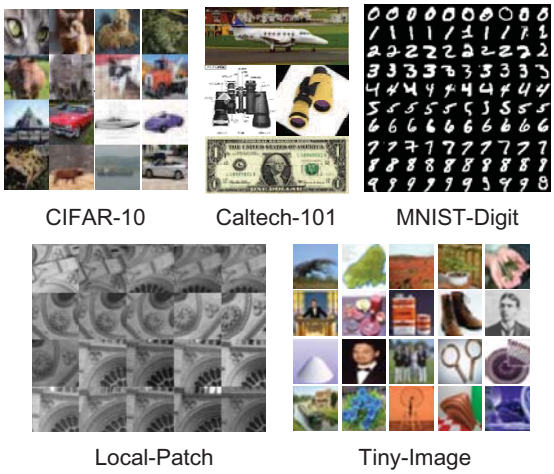

Figure 4. Example images used in the experiments.

In Figure 5 we plot the retrieval performance on Caltech-101, MNIST and CIFAR-10 using KLSH and the proposed LAMP. Following the evaluation scheme developed in [16], we collect all samples falling into hash buckets below a fixed un-normalized Hamming distance (2 in our experiments), and calculate the percentage of "good neighbors" (those having same labels with query sample). In case of fewer than 100 searched data, the threshold of Hamming distance will be increased until collecting enough nearest neighbors. By default two "similar" constraints are randomly generated for each data, resulting in 4K random constraints in all. It can be observed that the performance of our proposed LAMP outperforms pervious KLSH on all three databases. Performance superiority of LAMP is es-

[1] http://www.vision.caltech.edu/Image_Datasets/Caltech101/

[2] http://yann.lecun.com/exdb/mnist/

[3] http://www.cs.toronto.edu/kriz/cifar.html

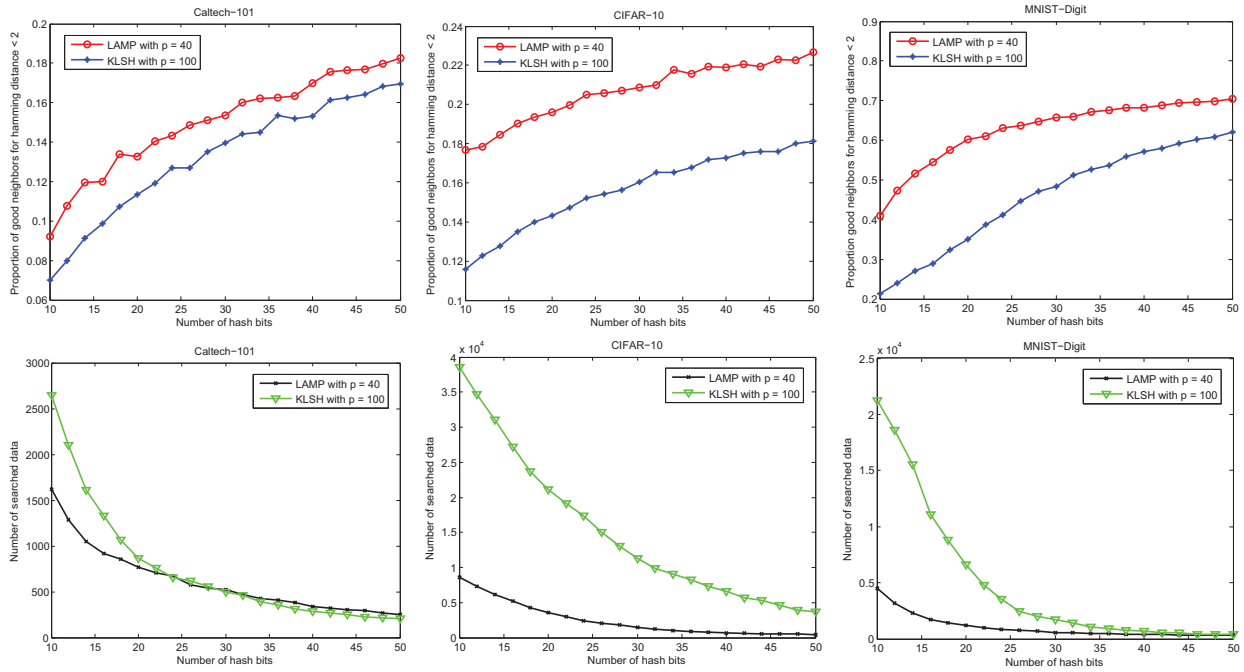

Figure 5. Hashing evaluation on Caltech-101, MNIST-Digit, and CIFAR-10. The first row shows the percentage of "good neighbor", while the second row illustrates the averaged retrieved sample numbers using 1K randomly-generated queries.

pecially obvious when the number of hash bits is small, in terms of both count of searched samples and percentage of good neighbors. When the number of hash bits continuously grows, KLSH tends to produce increasingly smaller pieces by cutting strongly-correlated data cliques. In contrast, LAMP tends to keep the integrity of those cliques, resulting in a more slowly and stably decreasing number of searched samples, as seen in the second row of Figure 5.

We also investigate the impact of different parameter choices in Figure 6. First, by raising the parameter of random constraint number per datum, a significant increase of retrieval accuracy can be observed, which is in accordance with the intention of "similar" constraints. In the second experiment, to validate the influence of side information, accuracies on CIFAR-10 under varying parameter λ_2 are presented. When $\lambda_2 = 0$, it is equivalent to not use side information and only rely on the max-margin criterion. It is observed that more side information (i.e., larger λ_2) brings better performance. Finally, we plot the performance curve under different p values. The performances are stable and slightly increase under larger p values, which proves LAMP's robustness to p.

5.2. Data Sets without Labels

Although learning with labels or tags becomes popular, such information is still missing in many vision benchmarks, or too noisy to use. Although the LAMP algorithm is not designed for such scenarios. However, it can still enhance hashing quality using the following trick: we can randomly generate a bunch of sample pairs that are among k-nearest-neighbors during linear scan, and impose them as "similar" constraints. In this way we conduct the evaluations on the following two data sets:

Local-Patch[4] is comprised of roughly 300K 32×32 subimages extracted from photos of Trevi Fountain (Rome), Notre Dame (Paris) and Half Dome (Yosemite). The goal is to evaluate fast algorithms which retrieve the corresponding patches given a query patch. We compute 128-d SIFT vector for each subimage, and utilize a uniform combination of three Gaussian kernels.

Tiny Image[5] consists of over 80 Million images crawled using Google's image search engine. Here we focus on a subset of Tiny Image, which contains roughly 1.7 Million images scaled to the resolution of 32×32 pixels. Following [9], 384-d GIST feature vectors are extracted and the similarity is measured through multiple Gaussian kernels.

Figure 7 shows the results. In each experiment, 2K samples are randomly selected, each of which contributes four similar constraints generated from its 4-nearest-neighbors. Each single hash function is working with $p = 100$ supporting vectors. Compared with unsupervised KLSH, the performance enhancement brought by the weak supervision and max-margin criterion is significantly observable on both data sets.

[4] http://phototour.cs.washington.edu/patches/default.htm
[5] http://people.csail.mit.edu/torralba/tinyimages/

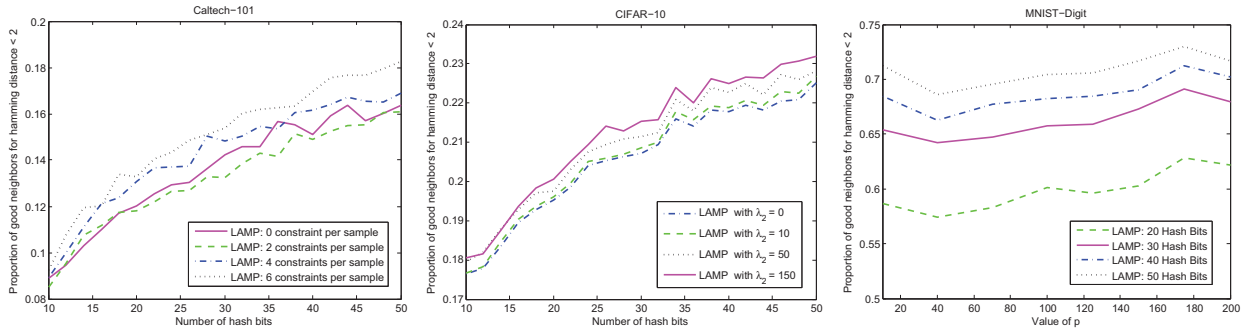

Figure 6. **Left**: the effect of different choices of constraint number per datum on Caltech-101. **Middle**: retrieval accuracy under different choices of parameter λ_2 on CIFAR-10. **Right**: performance given varying p values on MNIST-Digit.

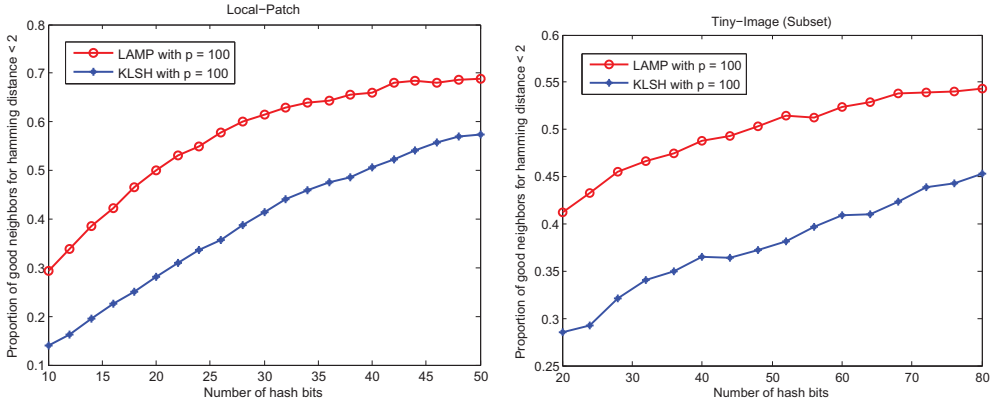

Figure 7. Experimental results on Local-Patch (left) and subset of Tiny-Image data sets (right).

6. Conclusions

We presented a novel hashing algorithm named LAMP, which generates high-quality hash functions with kernel tricks and weak supervision. The problem is formulated within a regularized maximum margin framework. Moreover, we provide a bound for the collision probability in the hash buckets based on Markov Random Fields theory. The proposed method makes no assumptions about the distribution of the input data, thus can be immediately applied to any image databases. The LAMP algorithm adopts a random sampling strategy in constructing both working set and supporting vectors, which enables it scalable for large-scale datasets. Empirical evaluations show its superiority over the state-of-the-art kernelized hashing algorithms.

Acknowledgment: the work in this paper is supported by NRF/IDM Program at Singapore, under research Grant NRF2008IDMIDM004-029.

References

[1] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Commun. ACM*, 51(1):117–122, 2008.

[2] B. Babenko, S. Branson, and S. Belongie. Similarity functions for categorization: from monolithic to category specific. In *ICCV*, 2009.

[3] J. Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, 1975.

[4] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[5] M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational geometry*, pages 253–262, 2004.

[6] Y. Hu, J. Wang, N. Yu, and X. Hua. Maximum margin clustering with pairwise constraints. In *ICDM*, 2008.

[7] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *STOC*, 1998.

[8] P. Jain, B. Kulis, and K. Grauman. Fast image search for learned metrics. In *CVPR*, 2008.

[9] B. Kulis and K. Grauman. Kernelized locality-sensitive hashing for scalable image search. In *ICCV*, 2009.

[10] B. Schölkopf, R. Herbrich, and A. Smola. A generalized representer theorem. In *COLT/EuroCOLT*, pages 416–426, 2001.

[11] B. Scholkopf and A. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT Press, Cambridge, MA, USA, 2001.

[12] G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter-sensitive hashing. In *ICCV*, 2003.

[13] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition. In *CVPR*, 2008.

[14] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

[15] A. Vishwanathan, A. Smola, and S. Vishwanathan. Kernel methods for missing variables. In *AISTAT*, 2005.

[16] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *NIPS*, 2008.

[17] A. Yuille and A. Rangarajan. The concave-convex procedure. *Neural Comput.*, 15(4):915–936, 2003.