LABEL PROPAGATION HASHING BASED ON P-STABLE DISTRIBUTION AND COORDINATE DESCENT

Haichuan Yang, Xiao Bai, Chuntian Liu*

School of Computer Science and Engineering
Beihang University
Haidian District, Beijing, China

Jun Zhou

School of Information and
Communication Technology
Griffith University
Nathan, QLD 4111, Australia

ABSTRACT

Hashing is a useful tool for contents-based image retrieval on large scale database. This paper presents an unsupervised data-dependent hashing method which learns similarity preserving binary codes. It uses p-stable distribution and coordinate descent method to achieve a good approximate solution for an acknowledged objective of hashing. This method consists of two steps. Firstly, it uses p-stable distribution properties to generate an initial partial hashing solution. Next, coordinate descent method is used to extend this partial solution to be complete. Our approach combines the advantages of both data-independent and data-dependent methods, which makes full use of the training data, requires reduced training time, and is easy to implement. Experiments show that our method outperforms several other state-of-the-art methods.

***Index Terms*—** Hashing, Image retrieval, p-stable distribution, Coordinate descent

1. INTRODUCTION

Many traditional image retrieval methods can not cope well with the increasing amount of image data to be processed in daily life. One typical example is the nearest neighbor (NN) search, which is a fundamental method in computer vision, machine learning and information retrieval. In NN search, features of an object are typically represented as a vector in R^d, and a distance metric is applied to measure the similarity between feature vectors. A most widely used distance measure is the Euclidean distance, which corresponds to the $l_2 - norm$ of vectors. Under this setting, the searching time of NN search increases linearly with the amount of the data. This is impractical for large scale database. When the dimensionality $d \leq 3$, this problem can be solved by indexing techniques based on space partitioning. However, in practice, most of the image feature descriptors, for example GIST [1] or SIFT [2], have much higher dimensionality. Then the performance of these methods degrades to exhaustive search [3].

*This work was supported in part by NSFC under Grant 61105002.

To break the bottleneck of NN search, approximate nearest neighbor (ANN) approaches have been proposed. The key idea is to find the approximate nearest neighbor rather than the exact one. Locality-sensitive hashing was introduced for this purpose [4] and has attracted lots of attention. Its basic objective is to map the original vector $v \in R^d$ to a binary string $y \in \{1, -1\}^r$, and to guarantee neighbours in the original space having similar binary codes in the hamming space.

Hashing based techniques mainly can be classified into two categories, data-independent hashing and data-dependent hashing, according to whether they make use of a training set to learn the hash function. One representative data-independent scheme is the locality-sensitive hashing based on p-stable distribution introduced in [5]. In this method, each code is produced by thresholding the data's projections on a vector w whose entries are independent identically distributed (i.i.d) variables with a p-stable distribution, e.g., a Gaussian distribution. It has some obvious advantages, for example, no training phase, good theoretical support, and compatibility with different data sets. However, data independence causes inefficiency of the binary representation. Consequently, the length of codes r has to be very large to reduce false positive rate, meanwhile, using multiple hash tables to maintain an acceptable recall rate. This increases the storage costs and degrades query efficiency in practice.

Data-dependent method leads to the opposite case. It learns hash functions that fit a unique training set. Like most machine learning methods, it can be divided into supervised and unsupervised categories based on whether the training data are labelled or not. Representative unsupervised hashing methods include spectral hashing (SH) [6], anchor graph hashing (AGH) [7], and binary reconstructive embedding (BRE) [8]. For supervised hashing, weakly-supervised hashing [9] and kernel-based supervised hashing [10] are two recent solutions. Data-dependent methods often generate compact codes and use single hash table, which is very feasible in practice.

The contribution of this paper is summarized as follows. Firstly, we show how to use p-stable distribution properties

to generate a partial hashing solution for a similarity preserving objective function, and prove the correctness of the partial solution. Secondly, we complete a partial solution using coordinate descent method, and develop a novel hashing method. Our approach can keep the advantage of locality-sensitive hashing (LSH) [5] based on p-stable distribution, and generate more efficient compact codes. Finally, we show superior performance of the proposed method over several state-of-the-art unsupervised hashing approaches in the experiments.

In the rest of the paper, we describe our method in Section 2, show experimental results in Section 3, and finally conclude and discuss the future work in Section 4.

2. LABEL PROPAGATION HASHING SCHEME

2.1. Similarity Preserving Objective of Hashing

The goal in this paper is to learn binary codes which satisfies the requirement that neighbors in the input space are mapped to similar codes in the Hamming space. Define $W_{ij} = exp(-\|x_i - x_j\|^2/\varepsilon^2)$ be the similarity between vectors v_i and v_j in the input space R^d, where ε is a distance control parameter. Our method seeks a r-bit Hamming embedding $Y \in \{1, -1\}^{n\times r}$ for n samples in the original space, and learns r hash functions $h : R^d \to \{1, -1\}$. If y_i represents binary code vector corresponding to the ith row of Y, we have an intuitive objective function:

$$\min \sum_{i,j} W_{ij}\|y_i - y_j\|^2 \tag{1}$$

$$subject\ to:\ y_i \in \{1, -1\}^r, \sum_i^n y_i = 0,$$

in which every column of Y should be independent of each other. Here, constraint $\sum_i^n y_i = 0$ allows data be mapped into hash table uniformly.

It has been proved in [6] that the above problem is N-P hard. Here we make some relaxation to get its approximate solution. Different from SH and AGH which relax it to an eigen decomposition problem, we achieve a good approximate solution using the p-stable distribution theory and the coordinate descent method. Following section introduces how to combine these two methods for our objective.

2.2. p-Stable Distribution

A random variable has a stable distribution if a linear combination of two independent copies of the variable has the same kind of distribution. A p-stable distribution D is a distribution that for any t real numbers $b_1...b_t$ and variables $X_1...X_t$ independent and identically drawn from distribution D, $\sum_i b_i X_i$ will have the same distribution as $(\sum_i \|b_i\|^p)^{1/p} X$, where $p\ (p \geq 0)$ is a parameter, and $X \sim D$ [11]. It has been proved in [12] that stable distribution exists when $p \in (0, 2]$. Particularly, when $p = 1$ and $p = 2$, the corresponding p-stable distributions are Cauchy distribution and Gaussian distribution.

p-stable distribution has been used for locality sensitive hashing. In [5], a locality-sensitive hashing method for the approximate nearest neighbor with l_p norm was presented. Although our approach is also partly based on p-stable distribution, we applied it in a novel way with training data. In the next section, some properties of p-stable distribution is analyzed, which leads to a partial hashing solution. Because our problem is based on Euclidean distance, only $p = 2$ is concerned here.

2.3. Initial Partial Solution

Here we show how to use p-stable distribution to get a partial hashing solution. Let the dimensionality of the original data be d, and w be a random vector with dimension d whose entries are generated independently from a standard Gaussian distribution with mean 0 and variance 1. Let v_i and v_j be two data vectors with dimension d, $c = w^T v_i - w^T v_j = w^T(v_i - v_j)$, then c will be distributed as $\|v_i - v_j\|G$, where G is a random variable following the standard Gaussian distribution, and w is a d-dimensional vector. Therefore, the difference between $w^T v_i$ and $w^T v_j$ follows a Gaussian distribution with mean 0 and variance $\|v_i - v_j\|^2$.

Let $(i, j) \in N$ if v_i and v_j are neighbors based on Euclidean distance (i,e, $\|v_i - v_j\|^2$ is very small) and $(i, j) \notin N$ be the opposite. When $(i, j) \in N$, $\|v_i - v_j\|^2$ is much smaller than when $(i, j) \notin N$. According to the probability density function of Gaussian distribution, if we choose an appropriate $\alpha(\alpha > 0)$, then in most situations, it can be guaranteed that

$$P(|c| > \alpha|(i, j) \notin N) \gg P(|c| > \alpha|(i, j) \in N) \tag{2}$$

By using Bayes' theorem, there is:

$$P((i, j) \notin N||c| > \alpha) = \frac{P(|c| > \alpha|(i, j) \notin N)P((i, j) \notin N)}{P(|c| > \alpha)} \tag{3}$$

$$P((i, j) \in N||c| > \alpha) = \frac{P(|c| > \alpha|(i, j) \in N)P((i, j) \in N)}{P(|c| > \alpha)} \tag{4}$$

In practice, $P((i, j) \notin N) > P((i, j) \in N)$, then it can be concluded:

$$P((i, j) \notin N||c| > \alpha) \gg P((i, j) \in N||c| > \alpha) \tag{5}$$

That is, if the difference between projections of two vectors on w is larger than α, there is high probability that they are not neighbors.

Recall our objective in (1). Firstly, we define a partial solution as a solution that contains only some y_i. For example, a partial solution may only have values of y_1, y_3, y_5. Secondly, we deal with each column of Y respectively, which means

we transform the original problem for $Y \in \{1,-1\}^{n\times r}$ to r problems for $Z_m \in \{1,-1\}^{n\times 1}(m=1,2...r)$, requiring that each Z_m be independent of each other.

Next, we generate a d-dimensional vector w whose entry is drawn independently from a Gaussian distribution. Let the vectors whose projections on w are far away from each other (larger than α) be set A. Considering that α is intractable, we do not choose the value of α, but choose a proportion of samples which have the smallest or the largest projections on w to form A. Then if $sign(w^T v_i) \neq sign(w^T v_j)$, there must be a corresponding α guaranteeing that $|w^T v_i - w^T v_j| > \alpha$. The proportion can be set empirically in the experiments.

For convenience, denoting the ith element in Z_m as $(Z_m)_i$. It can be easily proved that for $i \in A$, $(Z_m)_i = sign(w^T v_i)$ is a good approximate partial solution. If $\|(Z_m)_i - (Z_m)_j\|^2 > 0$, which means $sign(w^T v_i) \neq sign(w^T v_j)$, then $|w^T v_i - w^T v_j| > \alpha$, and in turn $\|v_i - v_j\|^2$ will be large with a high probability, which leads to $W_{ij} \approx 0$. When $\|(Z_m)_i - (Z_m)_j\|^2 = 0$, W_{ij} will not influence the sum.

Meanwhile, $w^T v_i$ follows a Gaussian distribution with zero mean. Because of the symmetry of its probability density function, the probability $P(w^T v_i > 0) = P(w^T v_i < 0)$ satisfies $\sum_i (Z_m)_i \approx 0$. Furthermore, the randomness of w makes $Z_m(m=1,2...r)$ be independent to each other.

2.4. Coordinate Descent Method

Once a partial hashing solution is available, it has to be extended to a complete solution. Let $z \in R^{n\times 1}$, $z_i = w^T v_i$ for $i \in A$ and $z_i = 0$ for $i \notin A$. For each $i \notin A$, z_i can be treated as the only variable while fixing other unknowns. Then we can minimize the objective function by letting the derivative with respect to z_i to be zero. This is a typical coordinate descent method. Then z_i can be updated iteratively via

$$z_i = \sum_{j=1, j\neq i}^{n} \frac{W_{ij} z_j}{\sum_{k=1,k\neq i}^{n} W_{ik}} \tag{6}$$

After several iterations, z will converge, then the binary code can be generated by thresholding z_i as $(Z_m)_i = sign(z_i)$ (for $i \notin A$).

Repeating this procedure r times, r $\{1,-1\}^{n\times 1}$ vectors can be generated. Finally, a matrix $Y = \{1,-1\}^{n\times r}$ can be produced by concatenating these vectors.

2.5. Binary Codes for Query

The above steps only generate the binary representation of the training set. The problem is then how to get the codes of a query. In SH [6], this problem was solved by an eigenfunction. Because our method is not based on eigen decomposition, we can not follow this method. Applying the same steps to the training set is another solution, but it is not more efficient than the exhaustive nearest neighbor search.

Our solution is as follows. Define function $f(v) = z$, it is obvious $f(v)$ is the desired function which maps origin vector v to soft label z. For efficiency purpose, we assume $f(v)$ to be linear, i.e., in the form of $f(v) = a^T v + b, a \in R^{d\times 1}, b \in R$, which can be solved as a linear regression model. Then thresholding $f(v)$ as $h_m = sign(f(v))$ leads to hash functions that generate hash codes for any input vector v.

The proposed method can be summarized in Algorithm 1. It should be noted that the process to solve Z_m is similar to a label propagation procedure, therefore we name it label propagation hashing (LPH). The partial solution for $i \in A$ is obtained using random projection and then propagated to $i \notin A$ by coordinate descent method.

Algorithm 1 Label propagation hashing

Input: training data $V \in R^{n\times d}$, similarity matrix W, length of hashing codes r

for $m = 1$ **to** r **do**
  generate random vector w: $w_i \sim Gaussian, i = 1...d$;
  compute $z \leftarrow Vw$;
  choose a proportion of the smallest and the largest entry in z, use their index to form A;
  $z_i \leftarrow 0$ when $i \notin A$;
  repeat
    for $i = 1$ **to** n **do**
      if $i \notin A$ **then**
        $z_i \leftarrow \sum_{j=1,j\neq i}^{n} \frac{W_{ij} z_j}{\sum_{k=1,k\neq i}^{n} W_{ik}}$;
      end if
    end for
  until z is converged
  $Z_m \leftarrow sign(z)$;
  $h_m \leftarrow$LinearRegression(z, V);
end for
$Y = [Z_1, Z_2...Z_r]$;

3. EXPERIMENTS

3.1. Evaluation Protocols and Baseline Methods

We evaluated the proposed method on a public CIFAR-10 data set [13]. It consists of 60000 32×32 color images in 10 classes, with 6000 images per class. The classes are airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. Figure 1 shows some images which are randomly selected from each class. We used a 384-dimensional GIST descriptor to represent each image.

Following the protocols used in many unsupervised hashing methods, we used Euclidean neighbors in the original space as the ground truth. Specifically, we set the average distance to the 50th nearest neighbor as the threshold to decide whether a sample is a true positive or not. Since our method (LPH) is unsupervised, three other unsupervised methods were selected as the baseline, which includ-

ed locality-sensitive hashing (LSH) [5], spectral hashing (SH) [6], and binary reconstructive embedding (BRE) [8]. We randomly selected 1000 samples as the query images, and left the remaining 59000 images as the database for constructing the hash table. The size of the training set depends on the standard parameter in different methods respectively. Because LSH is a data-independent approach, no training sample is needed. For SH, all samples in the database constitute the training set. For BRE, according to the setting in [8], 1000 samples are randomly selected from the database. For Our method, the training set consists of 3000 samples in the database.

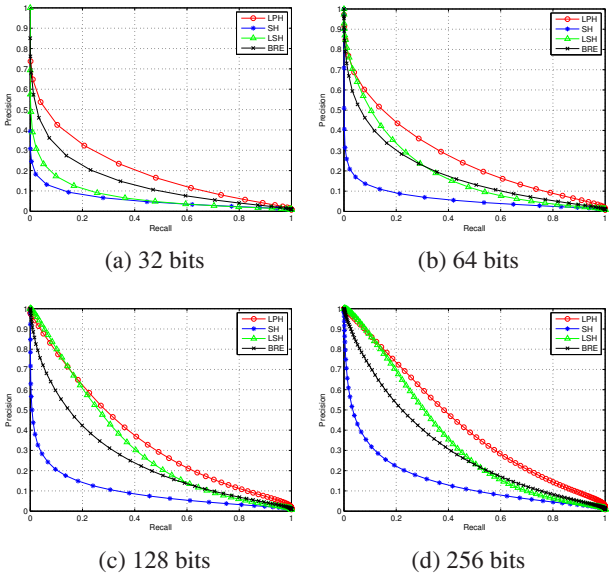
Fig. 1. Random samples in CIFAR-10 dataset. Each row contains 10 images of one class.

3.2. Performance under Different Code Length

The precision-recall curves with different code length are displayed in Figure 2. For each code length, the precision and recall were generated under different hamming radiuses. BRE has demonstrated better performance than LSH under 32 bits, but when the code length increases, LSH outperforms BRE. It is obvious that our method has consistently achieved better performance than the baselines.

3.3. Computational Cost

Table 1 shows the training time of each methods. Because LSH is data-independent, we do not list it here. All the experiments were implemented with MATLAB codes, ran on a PC with Core-i7 3.4GHZ CPU and 16GB memory. It can be seen



(a) 32 bits (b) 64 bits

(c) 128 bits (d) 256 bits

Fig. 2. Precision-recall curves on CIFAR-10.

that SH used the least time (only a few seconds). LPH used several minutes, and BRE used the longest time for training.

#bits	32	64	128	256
SH	1.12	1.28	1.63	2.37
LPH	65.78	93.84	148.382	237.83
BRE	108.54	839.18	1743.57	5023.13

Table 1. Training time (seconds) on CIFAR-10.

4. CONCLUSION

In this paper, we have reported a novel hashing method by combining the p-stable distribution theory with coordinate descent method. The effectiveness of this method was validated by experiments on the CIFAR-10 dataset. In the future, we would like to explore the application of this method to other scenarios such as clustering or dimensionality reduction.

5. REFERENCES

[1] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *International Journal of Computer Vision*, vol. 42, no. 3, pp. 145–175, 2001.

[2] D.G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[3] R. Weber, H.J. Schek, and S. Blott, "A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces," in *Proceedings of the International Conference on Very Large Data Bases*, 1998, pp. 194–205.

[4] P. Indyk and R. Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality," in *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, 1998, pp. 604–613.

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

[6] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Advances in neural information processing systems*, 2008, pp. 1753–1760.

[7] W. Liu, J. Wang, S. Kumar, and S.F. Chang, "Hashing with graphs," in *Proceedings of the 28th International Conference on Machine Learning*, 2011, pp. 1–8.

[8] B. Kulis and T. Darrell, "Learning to hash with binary reconstructive embeddings," *Advances in neural information processing systems*, vol. 22, pp. 1042–1050, 2009.

[9] Y. Mu, J. Shen, and S. Yan, "Weakly-supervised hashing in kernel space," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 3344–3351.

[10] W. Liu, J. Wang, R. Ji, Y.G. Jiang, and S.F. Chang, "Supervised hashing with kernels," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 2074–2081.

[11] P. Indyk, "Stable distributions, pseudorandom generators, embeddings and data stream computation," in *Proceedings. 41st Annual Symposium on Foundations of Computer Science*, 2000, pp. 189–197.

[12] V.M. Zolotarev, *One-dimensional stable distributions*, vol. 65, Amer Mathematical Society, 1986.

[13] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," *Master's thesis, Department of Computer Science, University of Toronto*, 2009.