

# Indexing Algorithm Based on Improved Sparse Local Sensitive Hashing

Zhu Yiwei

School of Electronic and Information Engineering, Zhejiang Business Technology Institute, Ningbo 315012, Zhejiang, China

**Abstract**—In this article, we propose a new semantic hashing algorithm to address the new-merging problems such as the difficulty in similarity measurement brought by high-dimensional data. Based on local sensitive hashing and spectral hashing, we introduce sparse principal component analysis (SPCA) to reduce the dimension of the data set which exclude the redundancy in the parameter list, and thus make high dimensional indexing and retrieval faster and more efficient. In the meanwhile, we employ Boosting algorithm in machine learning to determine the threshold of hashing, so as to improve its adaptive ability to real data and extend its range of application. According to experiments, this method not only has satisfying performance on multimedia data sets such as images and texts, but also performs better than the common indexing methods.

**Index Terms**—Indexing; Local Sensitive Hashing; Sparse; PCA

## I. INTRODUCTION

During its rapid development, the internet is changing our lives beyond imagination. As the internet intertwines with real life, the information it has created and stored is increasing exponentially. More and more people get involved in the generation and transmission of Internet information and more and more services are developed to make life convenient. All of these have dramatically increased the information content of the Internet. The amount of multimedia data, whether they are images, audios or videos, is so large that we can undoubtedly call them Mass Data [1-3]. In 2008, the index number of Google exceeded 1,000 billion; on October nth 2009, the amount of images in Flickr was over 4 billion; in November 2010, the speed of video update in Youtube reached 35G/min, and all these data are increasing astonishingly. How to find expected data among the increasing Mass Data rapidly and correctly has become one of the urgent problems. By solving this problem, the efficiency a lot of Internet application will be improved. The key to solve this problem is to optimize the current indexing and retrieval algorithms and develop new algorithms to improve the efficiency and accuracy of data indexing and retrieval. While multimedia data is increasing exponentially, they have caught some new features, the most obvious one of which is the high dimension of parameters. Low dimensional parameters no longer fully present multimedia data due to their rapid increase, and the previous monolithic expression cannot

accurately present the various kinds of multimedia data. Moreover, the application of high-dimensional and uniform expression to present various multimedia data has become the need of users and the goal of researchers.

To present multimedia data better and more comprehensively, the number of parameters need to be extracted reaches hundreds of thousands from them, whether texts, images, audios or videos. These high dimensional parameters not only made similarity measurement and semantic analysis difficult, but also place huge challenges to indexing and retrieval algorithms. Although traditional algorithms such as B-Tree, R-Tree [4] and K-DB Tree [5-6] can evolve into new algorithms that are application to high dimension, their efficiency decreases rapidly as the dimension increases. It is called Curse of Dimensionality that the efficiency of an algorithm in high dimension is lower than liner search while it is efficient in low dimension. It can be proved that the number of data to be retrieved  $N$  must satisfy the equation  $N \gg 2^k$ , where  $k$  is the dimension, other the efficiency of tree indexing is similar to or even lower than linear search. Although the multimedia data nowadays enters mass era, their amount dose not satisfy exponential growth as the data dimension increases. In addition, due to the limitations of block indexing, the efficiency of tree indexing is similar to linear search during the search of data points similar to keywords, so it cannot meet the need of quick retrieval in mass data. Thus, accurate tree indexing cannot satisfy the requirements brought by multimedia data indexing and retrieval under current Internet conditions, and the establishment of new methods to solve these problems has become more and more urgent.

New indexing and retrieval algorithms dealing with high dimensional parameters have emerged because of demand, in which semantic hashing [7] has become popular in researches due to its superior attributes and satisfying performance. Semantic Hashing is a hashing algorithm that maps high dimensional vectors to low dimensional Hamming Space, meanwhile maintain the similarity as in the original space, i.e. Hamming Distance in the new space reflects the similarity in the original space. Examples are Locality Sensitive Hashing (LSH) [8], Restricted Boltzmann Machine (RBM) [9], Parameter Sensitive Hashing (PSH) [10], and Spectral Hashing (SH) [11]. Semantic Hashing has phased in the concept of Approximate that users are more concerned about the

speed of search than the accuracy. In most cases, the exactly accurate search result is not necessary rather an approximate result can meet the most needs of users. Thus, via Hashing the data related to the query at a certain probability can be located from a dataset. If the accuracy of the result is somehow ensured, combined with the rapidity of similarity measurement in Hamming Space and the easiness to further the retrieval result, the efficiency of indexing and retrieval will be obviously improved.

Despite that many researchers at home and abroad have devoted to high dimensional indexing via Semantic Hashing and a number of effective algorithms have been developed, the following problems in high dimensional indexing remain: 1) the insufficiency of ability to process mass data. Although some machine learning algorithms showed good results in experiments, they suffer from limited ability to extend due to the complexity of the models. In reality, mass data exist apart from the training set and the incapability of the models to extend to data outside the training set leads to the decrease of indexing efficiency, thus the models are not applicable to real data indexing. 2) The incapability to deal with redundancy in high dimensional parameters. Algorithms such as SH leave out the problem of redundancy in the original vector space because they process the training data directory without pretreatment, which results in bad performance when the number of indexing result is small. On the other hand, the additional resource cost of mass data process in reality is large with every increased digit in the indexing result, which compromises the performance of these algorithms. 3) The lack of correlation between algorithms and data. Algorithms with well performed model such as LSH lack correlation with real data due to their completeness, which makes proper adjustment according to different datasets difficult. Thus these algorithms barely maintain satisfying performance on datasets from all areas, especially those with distinctive features and highly targeted.

In order to resolve the problems of existing high-dimensional indexing algorithm while deal with massive data, the paper focuses on semantics hash algorithm of high-dimension image indexing and presents a new indexing algorithm with more stronger applicability, more accuracy and efficiency, which combines improved locality sensitive hashing and sparse principal component analysis based on the work of predecessors. Each dimension of the raw data takes part in dimensionality reduction in the form of a linear combination during PCA procession. However, for a given training set, an original feature vector set can be obtained. A data point can be fully expressed via the only number of original features, especially the image data, which is related with a limited number of visual words. In this paper, a more reasonable method of dimension reduction is proposed which fully maintains the similarity of original feature space and excludes the redundant features in the original vectors, thus removing the affections of indexing efficiency brought from feature over-complete. And the solution can be obtained through steps of dimensions reduction and

mapping. In practical applications, most of the original data set does not meet the assumption of evenly distributed in multi-dimensional feature space and different data sets usually have different properties, so the optimal threshold often exists the only theoretical value. Obtaining threshold based on Boosting and the index via mapping results is another innovation of this paper. So the algorithm has good adaptability with different data sets, for the appropriate thresholds.

The article starts from semantic hashing and, via discussion on binary coding requirements and methods, theoretically obtain the best Semantic Hashing coding method. Meanwhile it takes into consideration parameter redundancy, extensive ability of models and adaptation to different datasets. After theoretical reduction, problem simplification and optimization and on the basis of traditional hashing, this article phases in Sparse Principal Component Analysis (SPCA), which forwards related global optimization algorithm combining with Boosting algorithm in machine learning and provides binary indexing algorithms with wide application range for different types of high dimensional mass data.

## II. RELATED WORKS

Traditional indexing algorithms such as R-Tree [4], KDB Tree [5] and B- Tree have good performance in low dimensional spaces, but their efficiency decreases rapidly as the dimension increases. For example, in R- Tree, a search starts from the root node, and traverses one by one the child nodes whose covering space contains query space, until it reaches leaf nodes. To insert a node, start a search from the root node for the node requiring the smallest extended space to contain the present keyword, until it reaches a leaf node. Then insert the keyword below this leaf node, and adjust the indexing structure similar as in B- Tree. To delete a node, perform a search at first, then delete the target leaf node and adjust the indexing structure as in B- Tree. R- Tree is a completely dynamic and highly balanced tree structure with high storage space usage. In R- Tree, search, insertion and deletion can be performed at the same time without periodic indexing rearrangement. However, in R- Tree indexing sub-spaces are allowed to overlap, and a high dimensional object can only be allocated into one sub-space, which results in variable search path. In addition, dynamic insertion leads to excessive number of space overlapping and dead-space, thus compromises the effectiveness of the algorithm.

Tree indexing structure is unable to solve Curse of Dimensionality, i.e. the complexity of the algorithm increases exponentially as the dimension of data increases. In addition, as the number of indexing data goes up, the tree indexing structure expand so rapidly that it becomes impossible to store it in the main memory. And if we store most of the structure in the auxiliary memory, the dispatch on auxiliary memory will further decrease the efficiency of tree indexing structure. New approaches to solve high dimensional indexing problems become more and more urgent, and this has intrigued thoughts about retrieval. Retrieval is the search of one or more data

points which is similar to the query words among a dataset, which is called Approximate. In real-life application, the exactly nearest neighbor as generated from tree indexing structure is not always necessary, especially in image indexing, where an approximate answer can meet the need of users in most cases. Therefore, indexing algorithms based on approximation have been developed.

LSH has provided an approach to solve nearest neighbor problems. If a hashing function  $H$  satisfies the following conditions: if data point  $p$  is the  $r$  neighbor of point  $q$ , then  $P(H(p)=H(q)) \geq p_1$ , otherwise  $P(H(p)=H(q)) \leq p_2$ , where  $p_1 \geq p_2$  and  $c > 1$ . Then  $H$  is called  $(c, r, p_1, p_2)$  sensitive hashing. And if we can find one sensitive hashing function,  $p$  and  $q$  are neighbors at certain probability. After the generations of traditional LSH, a series of location sensitive hashing have been forwarded, the most well-known of which is E2LSH [12]. E2LSH maps high dimensional parameters into low dimensional sub-spaces via linear mapping, maintains the sensitivity to locations and improves the accuracy compared to traditional LSH. Therefore it is widely used.

LSH algorithm has dramatically improved the efficiency of high dimensional indexing, and also provides a new approach to solve such problems. However, the random mapping, the prerequisite of this algorithm, inescapably leads to uncontrollability, i.e. the increase of indexing result number and the adjustment of algorithm parameters do not make much difference to the indexing result, which results in low correlation between the algorithm and the dataset and expected bad outcomes in some conditions. Furthermore, faced with the indexing need of mass data in modern times, the comparison of the distances between all candidate points and the query keywords is almost impossible. When the requirements on accuracy are stringent, LSH fails to ensure the accuracy and provide improved algorithm according to the features of the dataset.

To solve the problem of instability and uncontrollability of LSH random mapping, researchers have phased all kinds of machine learning method into Semantic Hashing and developed some algorithms that have higher specificity to datasets, of which RBM is the most representative algorithm. RBM is a deep machine learning model. First of all, it performs pre-training via vertical and hierarchical nonlinear learning to learn from original data upwards, until the generation of separate RBM Machine. In this way, high dimensional data points are converted to low dimensional data. Then, the eventual binary indexing result is reached via fine-tuning on parameters and weights.

Different from RBM, PSH applies relative simple learning method, Boosting. Boosting is a method to enhance the generalization of machine learning. By allocating different weights to a series of weak classifier, it adds up to a strong one. PSH applies Boosting to indexing computation converting Semantic Hashing coding problems into binary classification problems. This method gets the imbedded sub-space of the original space by similarity sensitive Boosting coding learning, and

measures the similarities between images after indexing via weighted Hamming distances. Further study has showed that this algorithm can evolve into more accurate Semantic Hashing which utilizes Hamming distance directory. It is shown by experiments and RBM and PSH all have better performance than LSH, which proves that the inclusion of machine learning into Semantic Hashing is a successful attempt. Although the complexity of PSH is acceptable and it has satisfying effect in many applications, it suffers from several problems. The most obvious problem is that PSH is develop to extract the pose from successive images rapidly, so it applies effectively to datasets that are clearly classified and with images sharing high similarity in the same class, but it fails to maintain good performance on mass image data from the Internet because the feature of the dataset is unclear.

### III. INDEXING ALGORITHM BASED ON IMPROVED LOCAL SENSITIVE HASHING

#### A. Local Sensitive Hashing

The idea of LSH algorithm is as follows: present high dimensional elements as points with coordinates, which are positive integers. Via a family of hashing functions  $\mathcal{F}$  all points in the space are mapped to  $n$  hash tables  $\mathcal{T}_i$ , in which  $n=|\mathcal{F}|$ , i.e. each hashing function  $f \in \mathcal{F}$  corresponds to a hash table and every hash table contains all points in the space. For each given inquiry factor  $q$ , calculate  $f_1(q)$ ,  $f_2(q)$ , ...,  $f_n(q)$ ,  $f_i(q) \in \mathcal{F}$ ,  $i=1, 2, \dots, n$ . Pick all points in the barrel of hash table  $\mathcal{T}_i$  which  $f_i(q)$  falls into and compare their distances to  $q$ , then pick  $K$  points with the smallest distances (K-NNS).

Local sensitive is generally defined as follows. If the following conditions are satisfied:

If  $p \in O(q, r_1)$ , then  $\Pr_{\mathcal{H}}(h(q)=h(p)) \geq p_1$

If  $p \notin O(q, r_2)$ , then  $\Pr_{\mathcal{H}}(h(q)=h(p)) \leq p_2$

Then function family  $\mathcal{H}$  mapping  $S$  to  $U$  is  $(r_1, r_2, p_1, p_2)$  sensitive to distance  $D$ .

#### B. Improved Sparse Local Sensitive Hashing

On the basis of the above study, Spectral Hashing rearranges the ideas of Semantic Hashing, and, combing the structure of Semantic Hashing with spectral analysis technique, attempts to get a Semantic Hashing from a new perspective. Under the prerequisite of reasonable analysis on coding conditions, Spectral Hashing performs spectral analysis to high dimensional samples. Then it calculates Eigen function, and, provided the dataset follows high dimensional homogeneous distribution, works out the hashing functions. The reduction of dimension is achieved via PCA, which converts high dimensional data to low dimensional ones whose different dimensions are uncorrelated [10]. The result can be further processed by binary indexing calculation. SH has improved the efficiency of image indexing, but two problems exist: during index coding, PCA is applied for dimension reduction, thus all high dimensional parameters are involved in the indexing result, and the

influence of redundancy remains; the dataset is assumed to follow homogeneous distribution, which is not the case in real life application, so its application is limited. For a given dataset  $S$ , select a training set consisting of  $N$  data points  $\{(x_i): i=1,2,\dots,N\}$ , in which  $x_i$  is the  $d$ -dimensional parameter vector of the  $i^{\text{th}}$  point and  $d$  is the number of parameters representing a data point.  $\Theta$  is the indexing function that maps  $d$ -dimensional parameter vector  $x_i$  to  $m$ -dimensional Hamming vector  $y_i$  from the original to Hamming space.  $\Theta$  is a Semantic Hashing, which means if  $x_i$  the vector presenting the  $i^{\text{th}}$  data point and  $x_j$  the vector presenting the  $j^{\text{th}}$  data point are close in the original space, then their corresponding vector after  $\Theta$ ,  $y_i$  and  $y_j$ , are also close. Therefore,  $\Theta$  can be defined as:

$$\Theta: x_i \in R^d \rightarrow y_i \in \{-1,1\}^m \quad (1)$$

If the Hamming vector is a binary one, this process is also called coding, and the indexing result is called coding result.

After defining the above vectors, the matrix in question is defined as  $X \in R^{N \times d}$ , which expresses the training set in the original space and the related indexing coding result is shown as  $Y \in \{-1,1\}^{N \times m}$ . The  $i^{\text{th}}$  row in  $X$  and  $Y$  presents the  $i^{\text{th}}$  data point in the original and hamming space respectively. In addition, some operations need to be defined in this article. Assume  $A$  is any matrix and  $a$  is a vector, then  $\text{Card}(a)$  is the cardinal number of  $a$ , i.e. the number of non-zero elements. Similarly,  $\text{Card}(A)$  is the cardinal number of  $A$ .  $\text{Rank}(A)$  is the rank of  $A$ ,  $\text{Cov}(A)$  the covariance matrix and, provided  $A$  is a square matrix,  $\text{Trace}(A)$  is the trace.

Knowing from the analysis on the requirements of Semantic Hashing indexing function, a good function  $\Theta$  should has the following properties: 1)  $\Theta$  fully maintains the similarity in the original space, i.e. the closer  $x_i$  and  $x_j$  is in the original space, the more probable their corresponding vectors  $y_i$  and  $y_j$  is close the Hamming space; 2)  $\Theta$  ensures effective indexing results, i.e. the original dataset  $S$  will be mapped to relatively low dimensional Hamming space, or provided similar exactness, relatively short coding length will fulfill the requirements; 3)  $\Theta$  has extensibility, i.e. once it is determined by the training set, it can be immediately applied to other dataset without high-frequency model recalculation; 4)  $\Theta$  can exclude the redundancy in the original vectors, i.e. for each data point, the indexing result can be presented only by some parameters in the original space that can discriminate its semantics, therefore the influence of redundant information caused by over-complete parameters is cut down.

Describe the above conditions except for 3) and 4), the target function can be shown as:

$$\begin{aligned} \min : & W(i, j) \|y_i - y_j\|^2 \\ \text{subject to : } & y_i \in \{-1,1\}^m, \sum_i y_i = 0, \frac{1}{N} \sum_i y_i y_i^T = I \end{aligned} \quad (2)$$

In formula (2),  $W(i, j)$  is the similarity between the original vectors  $x_i$  and  $x_j$ .  $y_i \in \{-1,1\}^m$  ensures condition 1)

that the indexing result is binary code.  $\sum y_j = 0$  ensures that each bit in the indexing result has equal probability of being -1 and +1.  $\frac{1}{N} \sum y_i y_i^T = I$  makes different bits of code are uncorrelated, which improves the efficiency of indexing and satisfies condition 2). However, the above formula is an NPC problem, so we give up the restrain of binary coding  $y_i \in \{-1,1\}^m$  and rewrite the formula by means of matrix:

$$\begin{aligned} \min : & \text{trace}(Y^T L Y) \\ \text{subject to : } & Y^T 1 = 0, Y^T Y = 1 \end{aligned} \quad (3)$$

In the above formula,  $L = D - W$  is a Laplace Matrix,  $W \in R^{N \times N}$  is the similarity matrix,  $D$  is an diagonal matrix whose diagonal elements are  $D(i,j) = \sum_i W(i,j)$ , and  $I$  is a unit matrix. The formula can be transformed to a Laplace Eigen map dimension reduction problem, and its solution is the eigenvectors in the Laplace Matrix  $L$  that corresponding to the  $m$  smallest eigenvalues (except for 0). The algorithm for Laplace Eigen map dimension reduction can execute dimension reduction in the original parameter space under similarity measurement method, but it cannot provide a uniform transmitting equation. This means that it cannot provide direct dimension reduction method for data outside the training set, and condition 3) is not satisfied. Moreover, the method itself cannot give correct boundary conditions to exclude redundancy, so condition 4) is not fulfilled. Therefore, solving one target function is not enough to satisfy all the requirements for a good indexing function, and it is necessary to divide the requirements into reasonable parts. In this article, we will divide the process into two steps, dimension reduction and mapping. Through the analysis of the previous 4 conditions, it is suggested that the accuracy of indexing result as in condition 1) is the basic requirement which must be followed in both steps. Condition 4) about redundancy restriction is the requirement on the original vectors, which should be the restriction in the dimension reduction step, while condition 2) about efficiency should be in the mapping step. At last, dualization of the mapping result of the threshold value will lead to the indexing result.

#### 1) Sparse PCA

The aim of this step is to reduce the dimension of the Eigen matrix of the original space  $X \in R^{N \times d}$  to the Eigen matrix  $B \in R^{N \times m}$ , provide condition 1) and 4), i.e. redundancy elimination and maintenance of similarity, are satisfied. Considering Euclidean space is the most popular eigenvector space, the original vector space is restricted in it and the inner product of vectors are used to measure their similarity, therefore Laplace Eigen map dimension reduction is converted to PCA:

$$\begin{aligned} \max : & p_i^T \text{cov}(X) p_i \\ \text{subject to : } & \|p_i\| = 1 \end{aligned} \quad (4)$$

Formula (4) gives the PCA version of target function (3), in which  $p_i$  is the principal component to be solved.

Related works have shown that formula (3) and (4) is equivalent when inner product is used to measure similarity in Euclidean space.

However, during the process of PCA [13], every dimension of the original data is involved in the dimension reduction process by means of linear combination. So for a given training set, the original eigenvector set is usually highly redundant. At most times, a data point can be fully presented by only several original parameters. For example in image indexing, an image is usually described by a limited number of words related to vision, like color related words is suitable to characterize rainbows and shape related words to cars. Include condition 4 to the target function and we can get a new one:

$$\begin{aligned} \max : & p_i^T \text{cov}(X) p_i - \rho \text{Card}^2(p_i) \\ \text{subject to : } & \|p_i\| = 1 \end{aligned} \quad (5)$$

In this formula, parameter  $p > 0$  controls the scarcity of the principal component  $p_i$ . This is an NPC problem and cannot be solved effectively. So we turn to convex optimization and rewrite the formula as [14]:

$$\begin{aligned} \max : & \text{Trace}(P \text{cov}(X)) - \rho \text{Card}(P) \\ \text{subject to : } & \text{Trace}(P) = 1 \\ & P \text{ is positive semidefinite} \\ & \text{Rank}(P) = 1 \end{aligned} \quad (6)$$

In this formula, matrix  $P = pp^T$ . Although this function is still difficult to solve, we can make the restriction less stringent and transform (6) to:

$$\begin{aligned} \max : & \text{Trace}(P \text{cov}(X)) - \rho \text{Card}(P) \\ \text{subject to : } & \text{Trace}(P) = 1 \\ & P \text{ is positive semidefinite} \end{aligned} \quad (7)$$

The detailed algorithm is shown in Algorithm 1.

Algorithm 1: Sparse PCA dimension reduction

Input: The targeted dimension  $m$  of the Eigen matrix  $X \in R^{N \times d}$ .

Output: Sparse PCA matrix  $M$ .

Step 1: Compute the covariance matrix of  $X$ ,  $\Sigma \in R^{d \times d}$ .

Step 2: Compute sparse principal component  $p$  of  $\Sigma$  via the positive semi-definite programming of formula (7).

Step 3: Update  $\Sigma$  according to the following equation.  
 $\Sigma = \Sigma - (p^T \Sigma p) p p^T$

Step 4: Repeat Step 2 and 3  $m$  times and obtain  $m$  sparse principal components  $\{p_1 \dots p_m\}$

Step 5: Set  $\{p_1 \dots p_m\}$  as column vectors of the sparse principal component matrix  $M \in R^{d \times m}$ .

The eigen matrix  $B \in R^{N \times m}$  after dimension reduction can be computed via matrix multiplication, i.e.  $B = X \times M$ . For any data point  $x$  in the same dataset  $S$ , its eigenvector  $b$  can be calculated through the same way.

The objective of binary indexing mapping is to find an efficient and extensible Semantic Hashing function, i.e. the function which maps matrix  $B$  to mapped matrix  $F$  and satisfies condition 1, 2 and 3 at the same time. Rewrite the description of data in expectation and convert

eigenvector to Eigen function, the target function can be written as:

$$\begin{aligned} \min : & \int \|f(x_1) - f(x_2)\|^2 W(x_1 - x_2) p(x_1) p(x_2) dx_1 dx_2 \\ \text{subject to : } & \int f(x) p(x) dx = 0 \\ & \int f(x) f(x)^T p(x) dx = I \end{aligned} \quad (8)$$

In this formula,  $f(x)$  is the mapping function,  $p(x)$  is the probability distribution and  $W(x_1, x_2)$  is the similarity between data points  $x_1$  and  $x_2$ . This function can be solved via the Eigen function of weighted Laplace-Beltrami operator. Weighted Laplace operator is defined as:

$$\frac{g(x)}{p(x)} = D(x) f(x) p(x) - \int W(s, x) f(s) p(s) ds \quad (9)$$

In this function,  $D(x) = \int W(x, s) ds$ . The solution of the above formula is the  $m$  smallest eigenvalues (except for 0) from the equation  $L_p f = \lambda f$ .

## 2) Threshold determination via boosting

The idea of threshold determination via boosting is described as follows: set every column of mapped matrix  $F$  as a binary linear weak classifier, get  $m$  of them whose error rates are the lowest via learning, and get the strong classifier with the lowest error rate via linear combination. On the basis of the theory stated previously, the similarity sensitive coding Boosting algorithm can be obtained intuitively: for each column from  $F$ , try to set one by one the data points from the current dataset as threshold and compute the error rate. Set the data point with the lowest error rate as the final threshold and thus the thresholds of all columns in the matrix are determined.

Algorithm 2: Threshold determination via Boosting

Input: Vectors to be determined  $F(:, n)$  and the neighboring matrix  $W_f \in R^{N \times N}$ .

Output:  $T_n$ , the threshold of  $F(:, n)$ .

Step 1: Compute  $N^2$  triples  $(F(i, n), F(j, n), W_f(i, j))$ ,  $i=1, \dots, N, j=1, \dots, N$ .

Step 2: Initialize the set  $A = \Phi$  and error count  $T_n = 0$ .

Step 3: For each triple  $(F(i, n), F(j, n), W_f(i, j))$

If  $F(i, n) > F(j, n)$ , then  $l_1 = 1$

If  $F(i, n) < F(j, n)$ , then  $l_1 = -1$

Else  $l_1 = 0$ ,  $l_2 = -l_1$  and  $A = A \cup \{(F(i, n), l_1, W_f(i, j)), (F(j, n), l_2, W_f(i, j))\}$

If  $W_f(i, j) = -1$ ,  $T_n = T_n + 1$ .

Step 4: Set the first element in set  $A$ , i.e. the value of  $F(:, n)$  as the keyword and sort the  $2N^2$  elements in  $A$  in ascending order.

Step 5:  $S_p = S_n = 0$ ,  $c_b = T_n$ ,  $T_n = \min(F(:, n)) - \text{eps}$

Step 6: For  $k=1: 2N^2$

$(f, l, w) = A[k]$

If  $w = 1$ ,  $S_p = S_p - l$

If  $w = -1$ ,  $S_n = S_n - l$

$c = T_n - S_n + S_p$

If  $c < c_b$ ,  $c_b = c$ ,  $T_j = f$

Apply Algorithm 2 to every column in matrix  $F$  we can get the corresponding thresholds, thus we can dualize them and obtain the eventual indexing result matrix  $Y$ .

#### IV. EXPERIMENTAL RESULTS

To test the effect of the method in this article, we designed a series experiments to show the performance of Sparse Spectral Hashing and other Semantic Hashing on image indexing. The experiments showed that the involvement of Boosting make Sparse Spectral Hashing more adaptive and effective in different types of image and text datasets.

##### A. Experimental Data Sets

###### 1) Image Data Sets

The experiment involved two image datasets, Paris and Oxford5k [16] [17]. First of all, we extracted the SIFT feature from the image datasets and perform K-means clustering, which generated the vision words related to the datasets. Each image was presented by a vector according to the number of related vision words.

Paris: this dataset consists of 6412 images from Flickr and most of them are buildings. We randomly selected 67 images as keywords for retrieval. To exclude the influence of the dimension of eigenvectors on the result, this dataset includes 300 vision words and are processed into 300-dimensional vectors.

Oxford5k: this dataset consists of 5062 images and the contents are 11 landmark buildings in Oxford University which contains 55 retrieval keywords (5 for each building). The rest images are grouped into training set. This is a public dataset commonly used in image indexing. There were 300 vision words and the dataset is processed into 300-dimensional vectors.

###### 2) Text Data Sets

This experiment involves the standard datasets of 20 Newsgroups [18]. At first, we extracted all words from the dataset and left out stop words. Then we computed the keywords through tf-idf and calculated the vectors formed by tf-idf of the keywords in every text.

20 Newsgroups: this dataset consists of 18445 passages of 20 topics, and there are 11315 left after processing (some of them only had titles or their content contained no keywords). The dataset was processed into 300-dimensional vectors. 4621 passages were selected as training set and 114 were treated as query set.

Reuters-21587: this dataset consists of 19043 passages of 91 topics. The dataset was processed into 300-dimensional vectors. 3879 passages were selected as training set and 95 were treated as query set.

Since Euclidean distance is the most popular measurement of similarity as well as the assumption under which the theory in this article was developed and it is application to all the comparison experiments, this article used Euclidean distance to measure the similarity and set the 1.5% of the average of Euclidean distance between the original parameters as the threshold of neighbor. Considering it is not possible to compare the Euclidean distances of all returned value when processing mass data and when the results of LSH indexing are the same, while enlarging the Hamming radius of returned data (the Hamming distance between the indexing results of a data point and a keyword) in mass data processing means greatly increasing the amount of data for post

processing (such as sorting the results), this experiment only returned the results with Hamming distance equals 0 between the indexing of dataset and the keywords.

The standards to evaluate the indexing results are F1 [14] and AUC [15]:

F1: Evaluate the results via precision and recall. Precision is the proportion of correct data in all returned data.

AUC: the size of area under Roc Curve. Its value is from 0 to 1. It is a popular evaluation in machine learning.

##### B. Experimental Analysis

The following tables show the results and analysis of various algorithms applied to various datasets. m is the number of digit of indexing coding. SSH is Sparse Spectral Hashing algorithm without Boosting and SPLSH with Boosting is the algorithm presented by the paper. LSH is the original Local Sensitive Hashing and E2LSH the linear mapping. RBM is Restricted Boltzmann Machine Semantic Hashing. PSH is Parameter Sensitive Hashing. SH is Spectral Hashing. Results for image datasets are:

Paris: To exclude the influence of the dimension of eigenvectors on the result, this dataset was preprocessed into 300-dimensional vectors. Results evaluated by F1 are shown in Table 1, in which the highlighted units are the algorithm with the highest score under the same number of digits of indexing coding.

TABLE I. RESULTS OF 300-DIMENSIONAL PARIS IMAGE SET EVALUATED VIA F1

| F1 of 300-dimensional Paris image set |        |        |        |        |        |        |        |
|---------------------------------------|--------|--------|--------|--------|--------|--------|--------|
| m                                     | LSH    | RBM    | PSH    | SH     | E2LSH  | SPLSH  | SSH    |
| 2                                     | 0.3412 | 0.3564 | 0.3978 | 0.3642 | 0.3142 | 0.5248 | 0.3437 |
| 4                                     | 0.3401 | 0.3564 | 0.4742 | 0.3656 | 0.2358 | 0.5461 | 0.3489 |
| 8                                     | 0.3478 | 0.3564 | 0.5105 | 0.3681 | 0.2251 | 0.5612 | 0.3542 |
| 16                                    | 0.3325 | 0.3564 | 0.3359 | 0.3704 | 0.0001 | 0.5816 | 0.3704 |
| 32                                    | 0.3243 | 0.3564 | 0.2974 | 0.4083 | 0.0000 | 0.6903 | 0.4673 |

TABLE II. RESULTS OF 300-DIMENSIONAL PARIS IMAGE SET EVALUATED VIA AUC

| AUC of 300-dimensional Paris image set |        |        |        |        |        |        |        |
|--|--------|--------|--------|--------|--------|--------|--------|
| m                                      | LSH    | RBM    | PSH    | SH     | E2LSH  | SPLSH  | SSH    |
| 2                                      | 0.4785 | 0.5156 | 0.6125 | 0.5274 | 0.5516 | 0.7301 | 0.5159 |
| 4                                      | 0.4779 | 0.5156 | 0.6954 | 0.5281 | 0.5265 | 0.7452 | 0.5161 |
| 8                                      | 0.5036 | 0.5156 | 0.6936 | 0.5376 | 0.5293 | 0.7881 | 0.5197 |
| 16                                     | 0.4691 | 0.5156 | 0.6013 | 0.5398 | 0.5211 | 0.7995 | 0.5756 |
| 32                                     | 0.4576 | 0.5156 | 0.6257 | 0.6424 | 0.5211 | 0.8201 | 0.6271 |

Results evaluated by AUC are shown in Table 2, in which the highlighted units are the algorithm with the highest score under the same number of digits of indexing coding.

Oxford5k dataset was preprocessed into 300-dimensional original eigenvectors. Results evaluated by F1 are shown in Table 3, in which the highlighted units are the algorithm with the highest score under the same number of digits of indexing coding.

Oxford5k dataset was preprocessed into 300-dimensional original eigenvectors. Results evaluated by AUC are shown in Table 4, in which the highlighted units are the algorithm with the highest score under the same number of digits of indexing coding.

The first breakthrough of high dimensional indexing in this article has made great progress and has many well developed algorithms. The algorithm in this article had good performance in experiments and can be viewed as general algorithm.

TABLE III. RESULTS OF OXFORD5K IMAGE SET EVALUATED VIA F1

| F1 of 300-dimensional Oxford5k image set |        |        |        |        |        |        |        |
|--|--------|--------|--------|--------|--------|--------|--------|
| m  | LSH    | RBM    | PSH    | SH     | E2LSH  | SPLSH  | SSH    |
| 2  | 0.2213 | 0.2207 | 0.2501 | 0.2102 | 0.2011 | 0.2917 | 0.2156 |
| 4  | 0.2195 | 0.2207 | 0.2797 | 0.2101 | 0.1765 | 0.2917 | 0.2155 |
| 8  | 0.2165 | 0.2208 | 0.4403 | 0.2876 | 0.1076 | 0.4029 | 0.3343 |
| 16                                       | 0.2011 | 0.2374 | 0.2905 | 0.3012 | 0.0000 | 0.4652 | 0.4081 |
| 32                                       | 0.1993 | 0.2184 | 0.1969 | 0.3517 | 0.0000 | 0.5278 | 0.5103 |

TABLE IV. RESULTS OF OXFORD5K IMAGE SET EVALUATED VIA AUC

| AUC of 300-dimensional Oxford5k image set |        |        |        |        |        |        |        |
|---|--------|--------|--------|--------|--------|--------|--------|
| m   | LSH    | RBM    | PSH    | SH     | E2LSH  | SPLSH  | SSH    |
| 2   | 0.5011 | 0.5064 | 0.5873 | 0.5107 | 0.5356 | 0.6678 | 0.5059 |
| 4   | 0.4974 | 0.5063 | 0.6231 | 0.5112 | 0.5325 | 0.6678 | 0.5042 |
| 8   | 0.7286 | 0.5065 | 0.7016 | 0.6592 | 0.5366 | 0.7990 | 0.7274 |
| 16  | 0.7742 | 0.5491 | 0.5925 | 0.6315 | 0.5113 | 0.8137 | 0.7359 |
| 32  | 0.7418 | 0.5127 | 0.5501 | 0.6925 | 0.5113 | 0.7936 | 0.7130 |

TABLE V. RESULTS OF 20 NEWSGROUPS TEXT SET EVALUATED VIA F1

| F1 of 20 Newsgroups text set |        |        |        |        |        |        |        |
|------------------------------|--------|--------|--------|--------|--------|--------|--------|
| m                            | LSH    | RBM    | PSH    | SH     | E2LSH  | SPLSH  | SSH    |
| 2                            | 0.2981 | 0.2927 | 0.2809 | 0.2876 | 0.2911 | 0.4015 | 0.2156 |
| 4                            | 0.2981 | 0.2927 | 0.2769 | 0.2871 | 0.2965 | 0.4015 | 0.2155 |
| 8                            | 0.2992 | 0.2927 | 0.2495 | 0.2871 | 0.2943 | 0.4015 | 0.3343 |
| 16                           | 0.2992 | 0.2927 | 0.2011 | 0.3345 | 0.2911 | 0.4638 | 0.5081 |
| 32                           | 0.2991 | 0.2927 | 0.1872 | 0.3988 | 0.2991 | 0.3023 | 0.5103 |

20 Newsgroups dataset was preprocessed into 300-dimensional original eigenvectors. Results evaluated by F1 are shown in Table 5, in which the highlighted units are the algorithm with the highest score under the same number of digits of indexing coding.

20 Newsgroups dataset was preprocessed into 300-dimensional original eigenvectors. Results evaluated by AUC are shown in Table 6, in which the highlighted units are the algorithm with the highest score under the same number of digits of indexing coding.

TABLE VI. RESULTS OF 20 NEWSGROUPS TEXT SET EVALUATED VIA AUC

| AUC of 20 Newsgroups text set |        |        |        |        |        |        |        |
|-------------------------------|--------|--------|--------|--------|--------|--------|--------|
| m                             | LSH    | RBM    | PSH    | SH     | E2LSH  | SPLSH  | SSH    |
| 2                             | 0.4235 | 0.4175 | 0.4425 | 0.4223 | 0.4222 | 0.6637 | 0.4945 |
| 4                             | 0.4235 | 0.4175 | 0.4479 | 0.4223 | 0.4265 | 0.6637 | 0.5122 |
| 8                             | 0.4301 | 0.4175 | 0.4258 | 0.4223 | 0.4209 | 0.6637 | 0.5122 |
| 16                            | 0.4301 | 0.4175 | 0.4237 | 0.5783 | 0.4165 | 0.7322 | 0.5244 |
| 32                            | 0.4300 | 0.4175 | 0.4214 | 0.6464 | 0.4398 | 0.5096 | 0.5175 |

TABLE VII. RESULTS OF REUTERS-21587 TEXT SET EVALUATED VIA F1

| F1 of Reuters-21587 text set |        |        |        |        |        |        |        |
|------------------------------|--------|--------|--------|--------|--------|--------|--------|
| m                            | LSH    | RBM    | PSH    | SH     | E2LSH  | SPLSH  | SSH    |
| 2                            | 0.2938 | 0.3827 | 0.2761 | 0.3376 | 0.2936 | 0.3993 | 0.3146 |
| 4                            | 0.2938 | 0.3731 | 0.2769 | 0.3662 | 0.2936 | 0.4057 | 0.3459 |
| 8                            | 0.2960 | 0.3800 | 0.2614 | 0.4398 | 0.2961 | 0.4543 | 0.3433 |
| 16                           | 0.2992 | 0.4211 | 0.1543 | 0.4957 | 0.2937 | 0.4843 | 0.3658 |
| 32                           | 0.2960 | 0.2975 | 0.0872 | 0.2893 | 0.2989 | 0.3987 | 0.3934 |

Reuters-21587 dataset was preprocessed into 300-dimensional original eigenvectors. Results evaluated by

F1 are shown in Table 7, in which the highlighted units are the algorithm with the highest score under the same number of digits of indexing coding.

Reuters-21587 dataset was preprocessed into 300-dimensional original eigenvectors. Results evaluated by AUC are shown in Table 8, in which the highlighted units are the algorithm with the highest score under the same number of digits of indexing coding.

TABLE VIII. RESULTS OF REUTERS-21587 TEXT SET EVALUATED VIA AUC

| AUC of Reuters-21587 text set |        |        |        |        |        |        |        |
|-------------------------------|--------|--------|--------|--------|--------|--------|--------|
| m                             | LSH    | RBM    | PSH    | SH     | E2LSH  | SPLSH  | SSH    |
| 2                             | 0.4977 | 0.6571 | 0.4934 | 0.5921 | 0.4974 | 0.6893 | 0.5675 |
| 4                             | 0.4977 | 0.6543 | 0.4935 | 0.6341 | 0.4974 | 0.6931 | 0.5755 |
| 8                             | 0.5123 | 0.6544 | 0.4937 | 0.6974 | 0.5021 | 0.7421 | 0.6102 |
| 16                            | 0.5121 | 0.7145 | 0.4925 | 0.7143 | 0.5020 | 0.7322 | 0.7357 |
| 32                            | 0.5121 | 0.5643 | 0.4991 | 0.5983 | 0.5020 | 0.6936 | 0.7190 |

Seen from the above results, in most cases the improved hashing algorithm forwarded in this article performs better than others under various preprocessing method and dimension of original eigenvectors. Its integrate evaluations are even better than those of other algorithms.

Compared to Local Sensitive Hashing alone, the hashing algorithm combined with Boosting performs better in most cases. This proves that the involvement of machine learning can enhance the correlation between the algorithm and the dataset and thus improve the performance of the algorithm.

## V. CONCLUSIONS

In this article we forwarded an improved Semantic Hashing indexing algorithm to solve problems in image indexing via Semantic Hashing. Via the study of coding requirements and approaches, we proposed the optimized coding requirement and combined it with machine learning. Eventually we developed an algorithm that is theoretically correct and widely applicable. We included Sparse PCA in traditional hashing to exclude the redundancy in parameters, and phased in Boosting from machine learning to make the algorithm more adaptive according to different datasets. The experiments proved that the algorithm proposed by this article performed better than other Semantic Hashing not only on image indexing but also on text indexing.

## REFERENCES

- [1] Cao J, Zhang Y, Song Y, et al, "MCG-WEBV: A benchmark dataset for web video analysis", *Beijing: Institute of Computing Technology*, vol. 10, pp. 324-334, 2009.
- [2] H. J. Zhang, J. H. Wu, D. Zhongtial, "An integrated system for content-based video retrieval and browsing", *Pattern Recognition*, vol. 30, pp. 64-655, 1997.
- [3] Datta R, Joshi D, Li J, et al, "Image retrieval: Ideas, influences, and trends of the new age", *ACM Computing Surveys*, vol. 40, no. 2, pp. 1-60, 2008.
- [4] Guttman A. R- trees, "a dynamic index structure for spatial searching", *Proceedings of the 1984 ACM SIGMOD international conference On Management of data. New York: ACM Press*, pp. 47-57, 1984.

- [5] Comer D, "Ubiquitous B-tree", *ACM Computing Surveys (CSUR)*, vol. 11, no. 2, pp. 121- 137, 1979.
- [6] Robinson J, "The KDB-tree:a search structure for large multidimensional Dynamic indexes", *Proceedings of the 1981 ACM SIGMOD international Conference on Management of data. New York: ACM Press*, pp. 10-18, 1981.
- [7] Salakhutdinov R, Hinton G, "Semantic hashing", *International Journal of APProximate Reasoning*, vol. 50, no. 7, pp. 969-978, 2009.
- [8] Gionis A, Indyk P, Motwanlr, "Similarity search in high dimensions via Hashing", *Proceedings of the 25th International Conference on Very Large Data Bases. San Franeiseo: Morgan Kaufmann Publishers*, pp. 518-529, 1999.
- [9] Torralba A, Fergus R, Weiss Y, "Small codes and large image data bases for recognition", *2008. IEEE Conference on Computer Vision and Patten Reeognition*, pp. 1-8, 2008.
- [10] Shakhovich, G Viola P, Darrell T, "Fast Pose estimation with parameter-Sensitive hashing", 2003.
- [11] Weissy, T Orralba A, Fergus R, "Spcetral hashing", *Advances in neural information processing systems*, vol. 21, pp. 1753-1760, 2009.
- [12] Datar M, Immorlica N, Indyk et al. "Loeality-sensitive hashing secheme Based on P-stable distributions", *Proceedings of the twentieth annual Symposium on Computational geometry*, pp. 253-262, 2004.
- [13] Bengioy, Delalleauo, Rouxn, et al, "Learning eigen functions link speetral embedding and kernel PCA ", *Neural Computation*, vol. 16, no. 10, pp. 2197-2219, 2004.
- [14] Jenaton R, Obozinskig, Bach F, " Struetured sparse Principal component analysis", *Journal of Maehine Learning Researeh-Proceedings Track*, vol. 9, pp. 366-373, 2010.
- [15] Bengio Y, Delalleau O, Roux N, et al, "Learning eigenfunctions links spectral embedding and kernel PCA", *Neural Computation*, vol. 16, no. 10, pp. 2197-2219, 2004.
- [16] Philbin J, Chum O, Isard M, et al, "Lost in quantization: Improving Particular object retrieval in large scale image data bases", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-8, 2008.
- [17] Philbin J, Chum O, Isard M, et al, "Objeet retrieval with large vocabularies and fast spatial matching", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-8, 2007.
- [18] Yang Y, Pedersen J, "A comparative study on feature seleetion intext categorization", *Proceeding of the 14 International conference on Machine Learning*, pp. 412-420, 1997.