Diverse Yet Efficient Retrieval using Locality Sensitive Hashing

Vidyadhar Rao
IIIT Hyderabad, India
vidyadhar.rao@research.iiit.ac.in

Prateek Jain
Microsoft Research India
prajain@microsoft.com

C.V Jawahar
IIIT Hyderabad, India
jawahar@iiit.ac.in

ABSTRACT

Typical retrieval systems have three requirements: a) Accurate retrieval, i.e., the method should have high precision, b) Diverse retrieval, i.e., the obtained set of samples should be diverse, and c) Retrieval time should be small. However, most of the existing methods address only one or two of the above mentioned requirements. In this work, we present a method based on *randomized* locality sensitive hashing which tries to address all of the above requirements simultaneously. While earlier hashing-based approaches considered approximate retrieval to be acceptable only for the sake of efficiency, we argue that one can further exploit approximate retrieval to provide impressive trade-offs between accuracy and diversity. We also extend our method to the problem of multi-label prediction, where the goal is to output a diverse and accurate set of labels for a given document in real-time. Finally, we present empirical results on image and text retrieval tasks and show that our method retrieves diverse and accurate images/labels while ensuring $100\times$-speed-up over the existing diverse retrieval approaches.

1. INTRODUCTION

Nearest neighbor (NN) retrieval [15, 17] is a critical sub-routine for machine learning, databases, and a variety of other disciplines. Basically, we have a database of points, and given an input query, the goal is to return the nearest point(s) to the query using some similarity metric. As a naïve linear scan of large databases is infeasible in practice, most of the research for NN retrieval has focused on making the retrieval efficient with either novel index structures [7, 39] or by approximating the distance computations [4, 19]. That is, the goal of these methods is: a) accurate retrieval, b) fast retrieval.

However in practice, NN retrieval methods [12, 21] are expected to meet one more criteria: diversity of retrieved data points. That is, it is typically desirable to find data-points that are diverse and cover a larger area of the space while maintaining high accuracy levels. For instance, when a user is looking for *flowers*, a typical NN retrieval system would tend to return all the images of the same flower (say *"lilly"*). But, it would be more useful to show a diverse range of images consisting of *sunflowers*, *lillies*, *roses*, etc. In this work, we propose a simple retrieval scheme that addresses all of the above mentioned requirements, i.e., a) good accuracy, b) fast retrieval time, and c) high diversity.

Our algorithm is based on the following simple observation: in most of the cases, one needs to trade-off accuracy for diversity. That is, rather than finding the nearest neighbor, we would need to select a point which is a bit farther from the given query but is *dissimilar* to the other retrieved points. Hence, we hypothesize that *approximate nearest neighbors can be used as a proxy to ensure that the retrieved points are diverse*. Figure 1 contrasts our approach with the different retrieval methods.

We propose a Locality Sensitive Hashing (LSH) [2, 8] based algorithm that guarantees approximate nearest neighbor retrieval in sub-linear time retrieval and superior diversity. We show that the effectiveness of our method depends on *randomization* in the design of the hash functions. Further, we modify the standard hash functions to take into account the distribution of the data for better performance. Our method retrieves points that are sampled uniformly at *random* to ensure diversity in the retrieval while maintaining reasonable number of relevant ones.

In our approach, it is easy to see that we can obtain higher accuracy with poor diversity and higher diversity with poor accuracy. Therefore, similar to precision and recall, there is a need to balance between accuracy and diversity in the retrieval. The later is important as there is a wide variety of diversity in what diversity means [14]. Evaluating the algorithms requires a measure that appropriately rewards diversity in the result list [29]. In this work, we keep a balance between accuracy and diversity and try to maximize these two criteria, simultaneously. We use harmonic mean between accuracy and diversity as the main performance measure. We believe that this performance measure is suitable for several applications i.e., the results would not change in essence with different notions of diversity and helps us empirically compare different methods. The major contributions of this paper are:

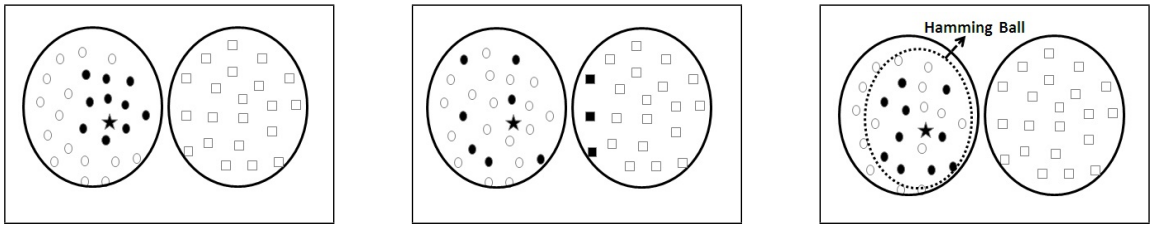
1. While approximate retrieval is acceptable only for the sake of efficiency, we argue that one can further exploit approximate retrieval to provide impressive trade-offs between accuracy and diversity. (*see Lemma 4.1 and Section 6*)

2. We propose hash functions that characterize the locality sensitive hashing to retrieve approximate nearest neighbors in sub-linear time with superior diversity. (*see Section 4*)

3. We extend the proposed method to diverse multi-label prediction problem and show that our method is not only orders of magnitude faster than the existing methods but also produces more accurate and diverse set of labels. (*see Section 5 and Section 8*)

ICMR'16, June 06-09, 2016, New York, NY, USA

DOI: http://dx.doi.org/10.1145/2911996.2911998



(a) Accurate, Not diverse **(b)** Not Accurate, diverse **(c)** Accurate, diverse

Figure 1: Consider a toy dataset with two classes: class A (○) and class B (□). We show the query point (⋆) along with ten points (●, ■) retrieved by various methods. Herein, we measure diversity to be the average pairwise distance between the retrieved set of points. a) A conventional similarity search method (e.g: k-NN) chooses points very close to the query and therefore, shows poor diversity. b) Greedy methods offer diversity but might make poor choices by retrieving points from the class B. c) Our method finds approximate nearest neighbors within a hamming ball of a certain radius around the query point and also ensures diversity among the points.

2. RELATED WORK

2.1 Optimizing Relevance and Diversity

Many of the diversification approaches [3, 5, 10, 32] are centered around an optimization problem that is derived from both relevance and diversity criteria. These methods can be broadly categorized into the following two approaches: (a) *Backward selection*: retrieve all the relevant samples and then rerank them using a clustering step to find a subset with high diversity [3, 10], (b) *Forward selection*: retrieve points sequentially by combining the relevance and diversity scores with a greedy algorithm [5, 16]. Most popular among these methods is MMR optimization [5] which recursively builds the result set by choosing the next optimal selection given the previous optimal selections. However, these methods result in high running times for massive databases.

Some studies attack the diversification problem in different ways. It was shown [31, 33] that natural forms of diversification arise via optimization of rank-based relevance criteria such as average precision and reciprocal rank. It is conjectured that optimizing $n\text{-}call@k$ metric correlates more strongly with diverse retrieval. More specifically, it is theoretically shown [31] that greedily optimizing expected $1\text{-}call@k$ w.r.t a latent subtopic model of binary relevance leads to a diverse retrieval algorithm that shares many features to the MMR optimization. Again, as it turns out, these methods render the task of obtaining diverse solutions extremely time-consuming and thus, are not scalable for several real-time tasks.

Complementary to all the above methods, our work ensures diversity in retrieval using randomization rather than optimization. In our work, instead of finding exact nearest neighbors to a query, we retrieve approximate nearest neighbors that are diverse. In our finding, we theoretically show that approximate NN retrieval via LSH naturally retrieves points that are diverse. Furthermore, our approach is quickly manageable for arbitrary databases of large sizes.

2.2 Application to Multi-label Prediction

A typical application of multi-label learning is automatic image/video tagging [6, 36], where the goal is to tag a given sample with all the relevant concepts/labels. The query is typically an instance (*e.g.*, image, text article) and the goal is to find the most relevant labels (*e.g.,* objects, topics). Moreover, one would like the labels to be diverse. For instance, for a given image, we would like to tag it with a small set of diverse labels rather than several very similar labels. However, the given labels are just some names and we typically do not have any features for the labels. E.g., for a given image of a lab, the appropriate tags might be chair, table, carpet, fan etc. In addition to the above requirement of accurate prediction of the positive labels (tags), we also require the obtained set of positive labels (tags) to be *diverse*. That is, for an image of a lab, we would prefer tags like {table, fan, carpet}, rather than tags like {long table, short table, chair}. Existing multi-label algorithms run in time linear in the number of labels which renders them infeasible for several real-time tasks [37, 40]; exceptions include random forest based methods [1, 28], however, it is not clear how to extend these methods to retrieve diverse set of labels.

To address this, we propose a method that extends our diverse NN retrieval based method to obtain diverse and sub-linear (in the number of labels) time multi-label prediction. Our method is based on the LEML method [40] which is an embedding based method. The key idea behind embedding based methods for multi-label learning is to embed both the given set of labels as well as the data points into a common low-dimensional space. The relevant labels are then recovered by NN retrieval for the given query point (in the embedded space). That is, we embed each label i into a d-dimensional space (say $y_i \in \mathbb{R}^d$) and the given test point is also embedded in the same space (say $x_q \in \mathbb{R}^d$). The relevant labels are obtained by finding y_i's that are closest to x_q. As the final prediction reduces to just NN retrieval, we can apply our method to obtain diverse set of labels in sub-linear time.(*see Section 5*)

3. OPTIMIZATION PROBLEM

Given a set of data points $\mathcal{X} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ where $x_i \in \mathbb{R}^d$, y_i is a label and a query point $q \in \mathbb{R}^d$, the goal is two-fold: a) retrieve a set of points $\mathcal{R}_q = \{x_{i_1}, \ldots, x_{i_k}\}$ such that a majority of their labels correctly predicts the label of q. b) The set of retrieved points $\mathcal{R}_q$ is "diverse". Note that in this work we are only interested in finding k points that are relevant to the query.

Although it is not quite clear on how relevance and diversity should be combined, we adopt a reminiscent [23] of the general paradigm in machine learning of combining loss functions that measures quality (e.g., training error, prior, or "relevance") and a regularization term that encourages desirable properties (e.g. smoothness, sparsity, or "diversity"). To this end, we define the following optimization problem:

$$\begin{aligned} \min\ & \lambda \Sigma_{i=1}^{n} \alpha_i \|q - x_i\|^2 - (1-\lambda) \Sigma_{ij} \alpha_i \alpha_j \|x_i - x_j\|^2 \\ \text{s.t.}\ & \Sigma_{i=1}^{n} \alpha_i = k; \forall i \in \{1, \ldots n\}\ \alpha_i \in \{0, 1\} \end{aligned} \tag{1}$$

where $\lambda \in [0, 1]$ is a parameter that defines the trade-off between the two terms, and α_i takes the value 1 if x_i is present in the result and 0 if it is not included in the retrieved result. Here, the first term measures the overall relevance of the retrieved set with respect

to the query. The second term measures the similarity among the retrieved points. That is, it penalizes the selection of multiple relevant points that are very similar to each other. By including this term in the objective function, we seek to find a set of points that are relevant to the query, but also dissimilar/diverse to each other.

Without loss of generality, we assume that x_i, q are normalized to unit norm, and with some simple substitutions like $\alpha = [\alpha_1, \ldots \alpha_n]$, $c = -[q^T x_1, \ldots, q^T x_n]$, G be gram matrix with $G_{ij} = x_i^T x_j$, the above objective is equivalent to

$$\begin{aligned} &\min \ \lambda c^T \alpha + (1-\lambda)\alpha^T G \alpha \\ &\text{s.t. } \alpha^T 1 = k; \alpha \in \{0,1\}^n \end{aligned} \tag{2}$$

From now on, we refer to the diverse retrieval problem in the form of the optimization problem in Eq.(2). Finding optimal solutions for the quadratic integer program in Eq.(2) is NP-hard [38]. Usually QP relaxations [30] (which are often called linear relaxations), where integer constraints are relaxed to interval constraints, are efficiently solvable. In this work, we consider the following simple approach[1] to solve Eq.(2): We first remove the integrality constraints on the variables i.e., allow variables to take on non-integral values and solve the relaxed optimization problem. Although, this method yields a good solution to Eq.(2) i.e., obtains accurate and diverse retrieval, solving the QP Relaxation is much more time consuming than the existing solutions (*see Table 1*).

To this end, the existing approaches i.e., greedy MMR optimization methods for Eq.(1) and the QP relaxation method for Eq.(2) suffer from three drawbacks: a) Running time of the algorithms is very high as it is required to recover several exact nearest neighbors. b) The obtained points might all be from a very small region of the space and hence the diversity of the selected set might not be large. c) Computation of the gram matrix may require an unreasonably large amount of memory overhead for large datasets. Therefore, it is of greatest interest to look for computationally efficient solutions for the diverse retrieval problem.

In this work, we propose an approach to overcome the above three issues based on the following high-level idea: instead of finding exact nearest neighbors, we perform *randomized approximate* nearest neighbor search using LSH which guarantees sub-linear time retrieval. We show that randomized hash functions selects points randomly around the query and hence are diverse, while still being relevant to the query.

4. METHODOLOGY

To find nearest neighbors, the basic LSH algorithm [8] concatenates a number of functions $h \in \mathcal{H}$ into one hash function $g \in \mathcal{G}$. Informally, we say that $\mathcal{H}$ is *locality-sensitive* if for any two points a and b, the probability that a and b collide under a random choice of hash function depends only on the distance between a and b. Several such families are known in the literature, see [2] for an overview.

DEFINITION 1. *(Locality Sensitive Hashing [2]): A family of hash functions $\mathcal{H} : R^d \rightarrow \{0,1\}$ is called (r, ϵ, P_1, P_2)-sensitive if for any $a, b \in R^d$*

$$\begin{cases} Pr_{h\in\mathcal{H}}[h(a) = h(b)] \geq P_1, & \text{if } d(a,b) \leq r \\ Pr_{h\in\mathcal{H}}[h(a) = h(b)] \leq P_2, & \text{if } d(a,b) \geq (1+\epsilon)r \end{cases}$$

Here, $\epsilon > 0$ is an arbitrary constant, $P_1 > P_2$ and $d(.,.)$ is some distance function.

[1]We refer to this method with QP-Rel in our experimental evaluations as one of our baselines.

In this work, we use ℓ_2 norm as the distance function and adopt the following hash function:

$$h(a) = sign(\rho \cdot a) \tag{3}$$

where $\rho \sim \mathcal{N}(0, I)$. It is well known that $h(a)$ is a LSH function w.r.t ℓ_2 norm and it is shown to satisfy the following [8]:

$$Pr(h(a) \neq h(b)) = \frac{1}{\pi} \cos^{-1}\left(\frac{a \cdot b}{\|a\|_2 \|b\|_2}\right). \tag{4}$$

Algorithm 1: LSH with random hash functions (LSH-Div)

Input: $\mathcal{X} = \{x_1 \ldots, x_n\}$, where $x_i \in R^d$, a query $q \in \mathbb{R}^d$ and k an integer.

1 **Preprocessing**: For each $i \in [1 \ldots L]$, construct a hash function, $g_i = [h_{1,i}, \ldots, h_{l,i}]$, where $h_{1,i}, \ldots, h_{l,i}$ are chosen at random from $\mathcal{H}$. Hash all points in $\mathcal{X}$ to the i^{th} hash table using the function g_i

2 $R \leftarrow \phi$

3 **for** $i \leftarrow 1$ ***to*** L **do**

4 Perform a hash of the query $g_i(q)$

5 Retrieve points from i^{th} hash table & append to $\mathcal{R}_q$

6 $\mathcal{S}_q \leftarrow \phi$

7 **for** $i \leftarrow 1$ ***to*** k **do**

8 $x^* \leftarrow \text{argmin}_{(x \in \mathcal{R}_q)}(\lambda \|q - x\|^2 - (1-\lambda)\Sigma_{s \in \mathcal{S}_q}\|x - s\|^2)$

9 $\mathcal{R}_q \leftarrow \mathcal{R}_q \setminus x^*$

10 $\mathcal{S}_q \leftarrow \mathcal{S}_q \cup x^*$

Output: $\mathcal{S}_q$, k diverse set of retrieved points for query q

We provide the details of our approach in the Algorithm 1: Given a query q, the algorithm executes in two phases: i) perform search through the hash tables, line(2-5), to report the approximate nearest neighbors, $R_q \subset \mathcal{X}$, and ii) perform k iterations, line(6-9), to report a diverse set of points, $S_q \subset R_q$. The essential control variables that maintain the trade-off between the accuracy and diversity of the retrieved points are: i) the number of points retrieved from hash table, $|R_q|$, and ii) the number of diverse set of points to be reported, k. Here, R_q can be controlled at the design of hash function, i.e., the number of matches to the query is proportional to $n^{\frac{1}{1+\epsilon}}$ (see Section 6). Therefore, line 7 is critical for the efficiency of the algorithm, since it is an expensive computation, especially when $|R_q|$ is very big, or k is large.

4.1 Diversity in Randomized Hashing

An interesting aspect of the above mentioned LSH function in Eq.(4) is that it is unbiased towards any particular direction, i.e., $Pr(h(q) \neq h(a))$ is dependent only on $\|q - a\|_2$ (assuming q, a are both normalized to unit norm vectors). But, depending on a sample hyper-plane $\rho \in \mathbb{R}^d$, a hash function can be biased towards one or the other direction, hence preferring points from a particular region. If the number of hash bits is large, then all the directions are sampled uniformly and hence the retrieved points are sampled uniformly from all the directions. *That is, the retrieval is not biased towards any particular region of the space.* We formalize the above observation in the following lemma.

DEFINITION 2. *(**Hoeffding Inequality** [25]) Let $Z_1, \ldots, Z_n$ be n i.i.d. random variables with $f(Z) \in [a, b]$. Then, with probability at least $1 - \delta$ we have*

$$P[\|\frac{1}{n}\sum_{i=1}^{n} f(Z_i) - E(f(Z))\|] \leq (b-a)\sqrt{\frac{\log(\frac{2}{\delta})}{2n}}$$

LEMMA 4.1. *Let $q \in \mathbb{R}^d$ and let $\mathcal{X}_q = \{x_1, \ldots, x_m\}$ be unit vectors such that $\|q - x_i\|_2 = \|q - x_j\|_2 = r$, $\forall i, j$. Let $p = \frac{1}{\pi}\cos^{-1}(1 - r^2/2)$. Also, let $r_1, \ldots, r_\ell \sim \mathcal{N}(0, I)$ be ℓ random vectors. Define hash bits $g(x) = [h_1(x) \ldots h_\ell(x)] \in \{0,1\}^{1\times\ell}$, where hash functions $h_b(x) = sign(r_b \cdot x)$, $1 \le b \le \ell$. Then, the following holds $\forall i$:*

$$p - \sqrt{\frac{\log(\frac{2}{\delta})}{2l}} \le \frac{1}{l}||g(q) - g(x_i)||_1 \le p + \sqrt{\frac{\log(\frac{2}{\delta})}{2l}}$$

That is, if $\sqrt{l} \gg 1/p$, then hash-bits of the query q are almost equi-distant to the hash-bits of each x_i.

PROOF. Consider random variable Z_{ib}, $1 \le i \le m$, $1 \le b \le \ell$ where $Z_{ib} = 1$ if $h_b(q) \ne h_b(x_i)$ and 0 otherwise. Note that Z_{ib} is a Bernoulli random variable with probability p. Also, Z_{ib}, $\forall 1 \le b \le \ell$ are all independent for a fixed i. Hence, applying Hoeffding's inequality, we obtain the required result. □

The above lemma shows that if $x_1, \ldots, x_m$ are all at distance r from a given query q then their respective hash bits are also at a similar distance to the hash bits of q. *That is, assuming randomized selection of the candidates from a hash bucket, probability of selecting any x_i is almost the same. That is, the points selected by LSH are nearly uniformly at random and are diverse.*

4.2 Randomized Compact Hashing

The conventional LSH approach [8] considers only random projections. Naturally, by doing random projection, we will lose some accuracy. But we can easily fix this problem by doing multiple rounds of random projections. However, we need to perform a large number of projections to increase the probability that similar points are mapped to similar hash codes. A fundamental result of the Johnson and Lindenstrauss Theorem [20] says that $O(\frac{\ln n}{\epsilon^2})$ random projections are needed to preserve the distance between any two pair of points, where ϵ is the relative error.

Using many random vectors to generate the hash tables (a long codeword), leads to a large storage space and a high computational cost, which would slow down the retrieval procedure. In practice, however, the data lies in a very small dimensional subspace of the ambient dimension and hence a random hyper-plane may not be very informative. Instead, we wish to use more data-driven hyper-planes that are more discriminative and separate out neighbors from far-away points. To this end, we obtain the hyper-planes ρ using principal components of the given data matrix. Principal components are the directions of highest variance of the data and capture the geometry of the dataset accurately. Hence, by using principal components, we aim to reduce the required number of hash bits and hash tables required to obtain the same accuracy in retrieval.

Precisely, given a data matrix $X \in \mathbb{R}^{d\times n}$ where i-th column of X is given by x_i, we obtain top-α principal components of X using SVD. That is, let $U \in \mathbb{R}^{d\times\alpha}$ be the singular vectors corresponding to the top-α singular values of X. Then, a hash function is given by: $h(x) = sign(\rho^T U^T x)$ where $\rho \sim \mathcal{N}(0, I)$ is a random α-dimensional hyper-plane. In the subsequent sections, we denote this algorithm using LSH-SDiv.

Many learning based hashing methods [22, 35] are proposed in literature. The simplest of all such approaches is PCA Hashing [34] which chooses the random projections to be the principal directions of the data directly. Our algorithm LSH-SDiv method is different from PCA Hashing in the sense that we still select random directions in the top components. Note that the above hash function has reduced randomness but still preserves the discriminative power by projecting the randomness onto top principal components of X. As shown in Section 8, the above hash function provides better nearest neighbor retrieval while recovering more diverse set of neighbors.

Algorithm 2: LSH based Multi-label Classification

Input: Train data: $\mathcal{X} = \{x_1, \ldots, x_n\}$, $\mathcal{Y} = \{y_1, \ldots, y_n\}$. Test data: $\mathcal{Q} = \{q_1, \ldots, q_m\}$. Parameters: α, k.

[W, H]=LEML($\mathcal{X}$, $\mathcal{Y}$, k);

$\mathcal{S}_q$ = LSH-SDiv($W, H^T q, \alpha$), $\forall q \in \mathcal{Q}$;

$\widehat{y}_q$ = Majority($\{y_i$ s.t. $x_i \in \mathcal{S}_q\}$), $\forall q \in \mathcal{Q}$;

Output: $\widehat{\mathcal{Y}}_\mathcal{Q} = \{\widehat{y}_{q_1}, \ldots, \widehat{y}_{q_m}\}$

5. DIVERSE MULTI-LABEL PREDICTION

We now present an extension of our method to the problem of multi-label classification. Let $\mathcal{X} = \{x_1, \ldots, x_n\}$, $x_i \in \mathbb{R}^d$ and $\mathcal{Y} = \{y_1, \ldots, y_n\}$, where $y_i \in \{-1, 1\}^L$ be L labels associated with the i-th data point. Then, the goal in the standard multi-label learning problem is to predict the label vector y_q accurately for a given query point q. In practice, the number of labels L is very large, so we require the diverse multi-label predictions to scale sub-linearly with L.

In this work, we build upon the LEML method proposed in [40] that can solve multi-label problems with a large number of labels and data points. In particular, LEML learns matrices W, H s.t. given a point q, its predicted labels is given by $y_q = sign(WH^T x)$ where $W \in \mathbb{R}^{L\times k}$ and $H \in R^{d\times k}$ and k is the rank of the parameter matrix WH^T. Typically, $k \ll \min(d, L)$ and hence the method scales linearly in both d and L with its prediction time being given by $O((d + L) \cdot k)$.

However, for several widespread problems, the $O(L)$ prediction time is quite large and makes the method infeasible in practice. Moreover, the obtained labels from this algorithm can all be very highly correlated and might not provide a diverse set of labels which we desire. We overcome both of the above limitations of the algorithm using the LSH based algorithm introduced in the previous section. We now describe our method in detail. Let $W_1, W_2, \ldots, W_L$ be L data points where $W_i \in \mathbb{R}^{1\times k}$ is the i-th row of W. Also, let $H^T x$ be a query point for a given x. Note that the task of obtaining α positive labels for given x is equivalent to finding α largest $W_i \cdot (H^T x)$. Hence, the problem is the same as nearest neighbor search with diversity where the data points are given by $\mathcal{W} = \{W_1, W_2, \ldots, W_L\}$ and the query point is given by $q = H^T x$.

We now apply our LSH based methods to the above setting to obtain a "diverse" set of labels for the given data point x. Moreover, the LSH Theorem in [13] shows that the time of retrieval is sub-linear in L which is necessary for the approach to scale to a large number of examples. See Algorithm 2 for the pseudo-code of our approach.

6. ALGORITHMIC ANALYSIS

Like most indexing strategies, LSH consists of two phases: *hash generation*, where the hash tables are constructed and *querying*, where the hash tables are used to look up for points similar to the query. Here, we discuss these two steps in more detail with respect to the suggested algorithms in the previous sections.

Accuracy: The data structure used by LSH scheme is randomized: the algorithm must output all points within the distance r from q, and can also output some points within the distance $(1+\epsilon)r$ from q. The algorithm guarantees that each point within the distance r from q is reported with a constant (tunable) probability. The

parameters l and L are chosen [18] to satisfy the requirement that neighbors are reported with a probability at least $(1-\delta)$. Note that the correctness probability is defined over the random bits selected by the algorithm, and we do not make any probabilistic assumptions about the data.

Diversity: In lemma 4.1, if the number of hash bits is large, i.e., if $\sqrt{l} \gg 1/p$, then hash-bits of the query q are almost equidistant to the hash-bits of each point in x_i. Then all the directions are sampled uniformly and hence the retrieved points are uniformly spread in all the directions. Therefore, for reasonable choice of the parameter l, the proposed algorithm obtains diverse set of points, $\mathcal{S}_q$ and has strong probabilistic guarantees for large databases of arbitrary dimensions.

Scalability: The time for evaluating the g_i functions for a query point q is $O(dlL)$ in general. For the angular hash functions chosen in our algorithm, each of the l bits output by a hash function g_i involves computing a dot product of the input vector with a random vector defining a hyperplane. Each dot product can be computed in time proportional to the number of non-zeros ζ rather than d. Thus, the total time is $O(\zeta lL)$. For an interested reader, see that the Theorem 2 of [8] which guarantees that L is at most $O(n^{\frac{1}{(1+\epsilon)}})$, where n denotes the total number of points in the database. Note that, in practice, the queries tend to follow the distribution of the data, and therefore, provide skewed accesses conforming to that of a data set.

7. EXPERIMENTAL SETUP

We demonstrate our approach applied to the following two tasks: (a) Image Category Retrieval and (b) Multi-label Prediction.

7.1 Evaluation Criteria

In both these experiments, our goal is two-fold: 1) improve diversity in the retrieval and 2) demonstrate speed-ups of our proposed algorithms. We measure the performance of our methods on three key aspects and characterize them in terms of the following metrics:

- **Accuracy:** We denote precision at k ($P@k$) as the measure of accuracy of the retrieval. This is the proportion of the relevant instances in the top k retrieved results.
- **Diversity:** For image retrieval, the diversity in the retrieved images is measured using entropy as $D = \frac{\sum_{i=1}^{m} s_i \log s_i}{\log m}$, where s_i is the fraction of images of i^{th} subcategory, and m is the number of subcategories for the category of interest. Sub-topic recall(SR) is measured as the fraction of subcategories retrieved for a given categorical query. For multi-label classification, the relationships between the labels is not a simple tree. It is better captured using a graph and the diversity is then computed using drank [26].
- **Efficiency:** Given a query, we consider retrieval time to be the time between posing a query and retrieving images/labels from the database. For LSH based methods, the hash tables are processed off-line, and so we do not consider the time spent to load the hash tables into the retrieval time. All the retrieval times are based on a Linux machine with Intel E5-2640 processor(s) and 96GB RAM.

7.2 Combining Accuracy and Diversity

Trade-offs between accuracy and efficiency in NN retrieval have been studied well in the past [4, 19, 39]. Many methods compromise on the accuracy for better efficiency. Similarly, emphasizing higher diversity may also lead to poor accuracy and hence, we want to formalize a metric that captures the trade-off between diversity and accuracy. To this end, we use (per data point) harmonic mean of accuracy and diversity as overall score for a given method (similar to f_{score} providing a trade off between precision and recall). That is, $h_{score}(\mathcal{A}) = \sum_i \frac{2 \cdot Acc(x_i) \cdot Diversity(x_i)}{Acc(x_i) + Diversity(x_i)}$, where $\mathcal{A}$ is a given algorithm and x_i's are given test points. In all of our experiments, parameters are chosen by cross validation such that the overall h_{score} is maximized.

8. EMPIRICAL RESULTS

8.1 Image Category Retrieval

For the image category retrieval, we consider a set of $42K$ images from ImageNet database [9] with 7 categories (*animal, bottle, flower, furniture, geography, music, vehicle*) with five subtopics for each. Images are represented as a bag of visual words histogram with a vocabulary size of $48K$ over the densely extracted SIFT [24] vectors. For each categorical query, we train an SVM hyperplane using LIBLINEAR [11]. Since, there are only seven categories in our dataset, for each category we created 50 queries by randomly sampling 16.67% of the images for training. After creating the queries (classifiers), we use the left over $35K$ images for the retrieval task.

We conducted two sets of experiments, 1) Retrieval without using hash functions and 2) Retrieval using hash functions, to evaluate the effectiveness of our proposed method. In the first set of experiments, we directly apply the existing diverse retrieval methods on the complete dataset. In the second set of experiments, we first select a candidate set of points by using the hash functions and then apply one of these methods to retrieve the images.

We hypothesize that using hash functions in combination with any of the retrieval methods will improve the diversity and the overall performance (h_{score}) with significant speed-ups. To validate our hypothesis, we evaluate various diverse retrieval methods in combination with our hash functions as described in Algorithm 1. It can be noted that lines 6-10 in Algorithm 1 can be replaced with various retrieval methods and can be compared against the methods without hash functions. In particular, we show the comparison with the following retrieval methods: the k-nearest neighbor (NN), the QP-Rel method and the diverse retrieval methods like Backward selection (Rerank), MMR [5].

In Table 1, we denote NH as Null Hash i.e., without using any hash function, LSH-Div with the random hash function and LSH-SDiv with the (randomized) PCA hash function. We report the quantitative results in Table 1 by the mean performance of all 350 queries. We can see in Table 1 that our hash functions in combination with various methods are superior to the methods with NH. Our extensions based on LSH-Div and LSH-SDiv hash functions outperform in all cases with respect to sub-topic recall (SR), diversity (D) and the h_{score}. Interestingly, LSH-Div and LSH-SDiv with NN report maximum h_{score} than any other methods. This observation implies that the approximate nearest neighbor search using standard LSH naturally favours diversity in the retrieval. This is in persistence to our result from Lemma 4.1. We also report a significant speed-up even for a moderate database of $35K$ images.

Readers familiar with LSH will also agree that our methods will enjoy better speed-up even in presence of larger databases and higher dimensional representations. In Table 3, we show a few qualitative results on this dataset with the top-10 images retrieved by different retrieval methods. Notice that the LSH-Div and LSH-SDiv methods consistently retrieve accurate and diverse set of images.

Table 1: We show the performance of various diverse retrieval methods on the ImageNet dataset. We evaluate the methods in terms of precision(P), sub-topic recall(SR) and Diversity(D) measures at top-10, top-20 and top-30 retrieved images. Numbers in **bold** indicate the top performers in SR, D, h_{score} and retrieval time. **NH** corresponds to the method without using any hash function. Notice that LSH-Div and LSH-SDiv hash functions consistently outperform the methods using NH in terms of h_{score}. In some cases, methods with NH show high h_{score} but our methods using hashing are fast in terms of retrieval time.

		precision at 10					**precision at 20**					**precision at 30**				
Method	**Hash Function**	P	SR	D	h_{score}	time (sec)	P	SR	D	h_{score}	time (sec)	P	SR	D	h_{score}	time (sec)
NN	NH	1.00	0.60	0.53	0.66	0.621	0.99	0.72	0.60	0.73	0.721	0.99	0.79	0.65	0.77	0.845
	LSH-Div	0.97	**0.79**	**0.76**	**0.84**	**0.112**	0.93	**0.93**	**0.86**	**0.89**	**0.137**	0.89	**0.98**	**0.91**	**0.90**	**0.179**
	LSH-SDiv	0.98	**0.76**	**0.73**	**0.83**	**0.181**	0.95	**0.89**	**0.85**	**0.89**	**0.183**	0.92	**0.95**	**0.89**	**0.90**	**0.106**
Rerank	NH	1.00	0.73	0.69	0.81	0.804	0.99	0.79	0.70	0.81	0.793	0.99	0.88	0.77	0.86	0.901
	LSH-Div	0.93	**0.80**	**0.76**	**0.83**	**0.142**	0.92	**0.93**	**0.86**	**0.88**	**0.146**	0.87	**0.98**	**0.90**	**0.88**	**0.214**
	LSH-SDiv	0.95	**0.79**	**0.76**	**0.84**	**0.154**	0.94	**0.91**	**0.85**	**0.89**	**0.179**	0.90	**0.95**	**0.88**	**0.89**	**0.203**
MMR	NH	0.95	0.75	0.71	0.80	5.686	0.98	0.86	0.77	0.85	11.193	0.97	0.90	0.80	0.87	17.162
	LSH-Div	0.91	**0.77**	**0.73**	**0.80**	**1.135**	0.91	**0.92**	**0.85**	**0.87**	**2.378**	0.87	**0.97**	**0.89**	**0.88**	**3.828**
	LSH-SDiv	0.92	**0.78**	**0.75**	**0.81**	**1.102**	0.93	**0.91**	**0.84**	**0.88**	**2.085**	0.89	**0.96**	**0.88**	**0.88**	**4.106**
QP-Rel	NH	1.00	0.74	0.69	0.81	704.9	1.00	0.82	0.73	0.84	947.09	1.00	0.87	0.76	0.86	1137.19
	LSH-Div	0.93	**0.80**	**0.77**	**0.83**	**0.487**	0.92	**0.94**	**0.86**	**0.88**	**0.499**	0.86	**0.98**	**0.90**	**0.88**	**0.502**
	LSH-SDiv	0.97	**0.78**	**0.74**	**0.83**	**0.447**	0.96	**0.89**	**0.82**	**0.88**	**0.464**	0.93	**0.95**	**0.86**	**0.89**	**0.473**

8.2 Multi-label Prediction

We use one of the largest multi-label datasets, LSHTC [27], to show the effectiveness of our proposed method. This dataset contains the Wikipedia documents with more than $300K$ labels. Since, the dataset is highly skewed, we selected only the documents which have at least 4 or more labels. Thus, we have a dataset of $754K$ documents with $259K$ unique labels. For our experiment, we randomly divide the data in 4:1 ratio for training and testing respectively. We use the large scale multi-label learning (LEML) [40] algorithm to train a linear multi-class classifier. This method is shown to provide state of the art results on many large multi-label prediction tasks.

In LSHTC3 dataset, the labels are associated with a category hierarchy which is cyclic and unbalanced, i.e., both the documents and subcategories are allowed to belong to more than one other category. In such cases, the notion of diversity, i.e., the extent to which the predicted labels belong to multiple categories can be estimated using drank [26]. If the labels belong to the same/few categories, it is considered to have low diversity. Relatively, if the labels belong to different categories, it is considered to have more diversity. Since, the category hierarchy graph is cyclic, we prune the hierarchy graph to obtain a balanced tree by using the Breadth First Search (BFS) traversal. Diversity of the predicted labels is computed as the drank score on this balanced tree. Accuracy for the predictions is computed as Precision, Recall and f_{score}.

Since, the number of labels for each document varies, we used a threshold parameter to limit the number of predicted labels to the documents. We select this threshold by cross validating such that it maximizes the f_{score}. It turns out that all methods produce a small number of diverse label predictions, which results in low recall. These low recall scores affect the f_{score}, which are consequently low as well. This is mainly due to the restrictive nature of these algorithms, as evidenced by the number of predicted labels, being very low for all methods. To quantify the over all performance of these methods, we compute the h_{score} as a harmonic mean of precision and drank score. We demonstrate that the proposed algorithm is effective and robust, since, it improves diversity even when retrieving relevant labels is difficult.

Table 2: Results on LSHTC3 challenge dataset with LEML, MMR, PCA-Hash, LSH-Div and LSH-SDiv methods. LSH-SDiv method significantly outperforms all other methods in terms of overall performance, h_{score} as well as the retrieval time.

Method	P	R	f_{score}	D	h_{score}	time (msec)
LEML [40]	0.304	0.196	0.192	0.827	0.534	137.1
MMR [5]	0.275	0.134	0.175	0.865	0.418	458.8
LSH-Div	0.144	0.088	0.083	0.825	0.437	7.2
PCA-Hash	0.265	0.096	0.121	0.872	0.669	5.9
LSH-SDiv	0.318	0.102	0.133	**0.919**	**0.734**	**5.7**

In Table 2, we report the performance of label prediction with LEML and MMR [5], and the proposed methods that predict diverse labels efficiently. As can be seen, the LSH-Div method shows a reasonable speed-up but fails to report many of the accurate labels, i.e., has low precision. The LSHTC3 dataset is highly sparse in a large dimension (20L), random projections generated by LSH-Div method are a bit inaccurate and have resulted in poor accuracy. In contrast, the LSH-SDiv method can successfully preserve the distances, i.e., reports accurate labels by projecting onto a set of principal components of the data while still producing diverse labels. Clearly, LSH-SDiv based hash function improves the diversity within the labels and outperforms LEML, MMR, PCA-Hash and LSH-Div methods in terms of h_{score}.

Figure 2 illustrates the performance of proposed method with respect to the parameter ϵ. In the figure, we show the performance obtained when 100 random projections are selected for the LSH-Div method. Notice that the conventional LSH fails to retrieve accurate labels and it requires a large number of random projections to retrieve accurate labels, which would slow down the retrieval procedure. For the LSH-SDiv method we project the 100 random projections onto the top-200 singular vectors obtained from the data. Moreover, LSH-SDiv typically requires much smaller number of hash functions than the standard LSH method and hence, is much faster as well. In summary, LSH-SDiv methods produces accurate as well as diverse labels, and we obtain a speed-up greater than 20 over LEML method and greater than 80 over MMR method.

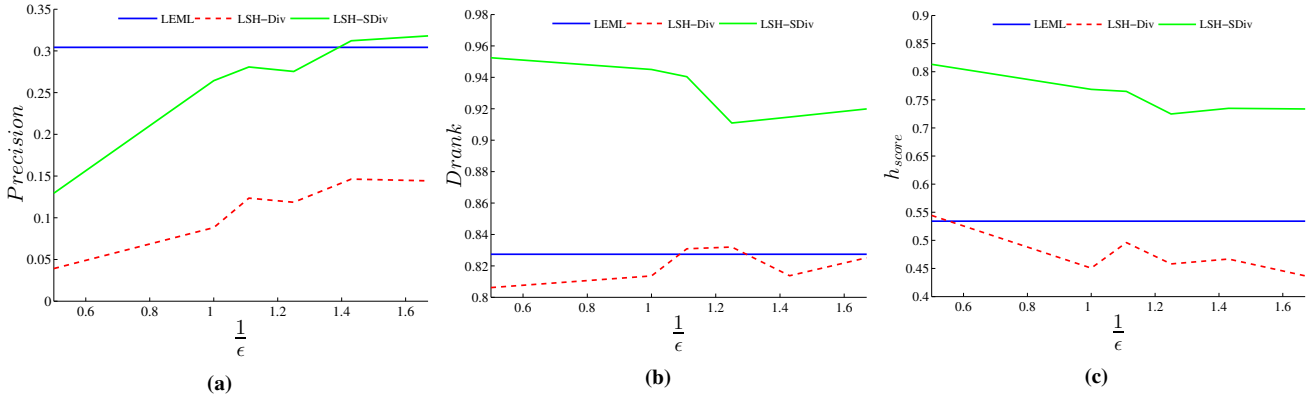


Figure 2: Sensitivity analysis of LSH based methods with respect to the parameter ϵ. We use LSHTC3 dataset for our study. (a) LSH-SDiv method gives better precision than LSH-Div method. (b) LSH-SDiv method shows better diversity than LEML and LSH-Div methods. (c) LSH-SDiv method performs significantly better than the LEML and LSH-Div methods in terms of h_{score}. (*Figure best viewed in color*)

9. CONCLUSIONS

In this paper, we present an approach to efficiently retrieve diverse results based on *randomized* locality sensitive hashing. We argue that standard hash functions retrieve points that are sampled uniformly at random in all directions and hence ensure diversity by default. We show that, for two applications (image and text retrieval), our methods retrieve significantly more diverse and accurate data points, when compared to the existing methods. Our results are appealing: a good balance between accuracy and diversity is obtained by using only a small number of hash functions. We obtained $100\times$-speed-up over existing diverse retrieval methods while ensuring high diversity in retrieval. The proposed solution is highly efficient with theoretical guarantees for sub-linear retrieval time and therefore, the algorithms are interesting and should be very useful and attractive for all practical purposes.

10. REFERENCES

[1] R. Agrawal, A. Gupta, Y. Prabhu, and M. Varma. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In *WWW*, 2013.

[2] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *FOCS*, 2006.

[3] T. Aytekin and M. Ö. Karakaya. Clustering-based diversity improvement in top-n recommendation. *IIS*, 2014.

[4] R. Basri, T. Hassner, and L. Zelnik-Manor. Approximate nearest subspace search. *TPAMI*, 2011.

[5] J. Carbonell and J. Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*, 1998.

[6] G. Carneiro, A. B. Chan, P. J. Moreno, and N. Vasconcelos. Supervised learning of semantic classes for image annotation and retrieval. *TPAMI*, 2007.

[7] L. Cayton and S. Dasgupta. A learning framework for nearest neighbor search. In *NIPS*, 2008.

[8] M. S. Charikar. Similarity estimation techniques from rounding algorithms. In *STOC*, 2002.

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[10] T. Deselaers, T. Gass, P. Dreuw, and H. Ney. Jointly optimising relevance and diversity in image retrieval. In *CIVR*, 2009.

[11] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *JMLR*, 2008.

[12] G. Ference, W.-C. Lee, H.-J. Jung, and D.-N. Yang. Spatial search for k diverse-near neighbors. In *CIKM*, 2013.

[13] A. Gionis, P. Indyk, and R. Motwani. Similarity Search in High Dimensions via Hashing. In *VLDB*, 1999.

[14] P. B. Golbus, J. A. Aslam, and C. L. Clarke. Increasing evaluation sensitivity to diversity. *Information Retrieval*, 2013.

[15] X. Han, S. Li, and Z. Shen. A k-nn method for large scale hierarchical text classification at lshtc3. *ECML PKDD*, 2015.

[16] J. He, H. Tong, Q. Mei, and B. Szymanski. Gender: A generic diversified ranking algorithm. In *NIPS*, 2012.

[17] M. E. Houle, X. Ma, V. Oria, and J. Sun. Improving the quality of k-nn graphs for image databases through vector sparsification. In *ACM ICMR*, 2014.

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

[19] P. Jain, S. Vijayanarasimhan, and K. Grauman. Hashing hyperplane queries to near points with applications to large-scale active learning. In *NIPS*, 2010.

[20] W. B. Johnson and J. Lindenstrauss. Extensions of lipschitz mappings into a hilbert space. 1984.

[21] O. Kucuktunc and H. Ferhatosmanoglu. λ-diverse nearest neighbors browsing for multidimensional data. *TKDE*, 2013.

[22] B. Kulis and T. Darrell. Learning to hash with binary reconstructive embeddings. In *NIPS*, 2009.

[23] H. Lin and J. Bilmes. A class of submodular functions for document summarization. In *HLT*. ACL, 2011.

[24] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004.

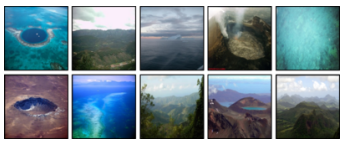
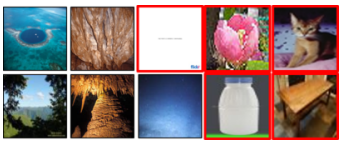
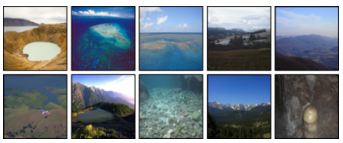
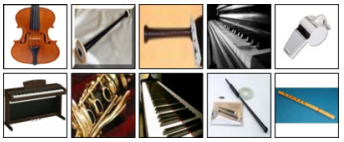
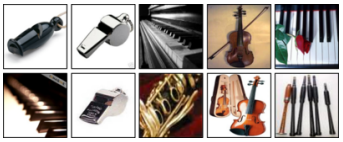
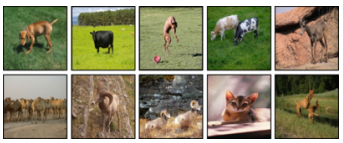
[25] G. Lugosi. Concentration-of-measure inequalities. 2004.

[26] P. K. R. Mittapally Kumara Swamy and S. Srivastava. Extracting diverse patterns with unbalanced concept hierarchy. In *PAKDD*, 2014.

[27] I. Partalas, A. Kosmopoulos, N. Baskiotis, T. Artieres, G. Paliouras, E. Gaussier, I. Androutsopoulos, M.-R. Amini, and P. Galinari. Lshtc: A benchmark for large-scale text classification. *arXiv preprint arXiv:1503.08581*, 2015.

[28] Y. Prabhu and M. Varma. Fastxml: a fast, accurate and stable tree-classifier for extreme multi-label learning. In *KDD*, 2014.

Table 3: Top-10 images retrieved for sample queries from the ImageNet database are shown: the first coloumn with the simple NN method, the second coloum with method based on MMR optimization, and the third coloumn with the proposed LSH-SDiv method. Notice that the greedy method fails to retrieve accurate images for some of the queries. Images marked with red box are incorrectly retrieved ones. Our method, consistently retrieves relevant images and simultaneouly shows better diversity. (*Image best viewed in color*)

	Simple NN	MMR	LSH-SDiv
Flower			
Vehicle			
Geography			
Furniture			
Bottle			
Musical Instrument			
Animal			

[29] F. Radlinski, P. N. Bennett, B. Carterette, and T. Joachims. Redundancy, diversity and interdependent document relevance. In *ACM SIGIR*, 2009.

[30] P. Ravikumar and J. Lafferty. Quadratic programming relaxations for metric labeling and markov random field map estimation. In *ICML*, 2006.

[31] S. Sanner, S. Guo, T. Graepel, S. Kharazmi, and S. Karimi. Diverse retrieval via greedy optimization of expected 1-call@k in a latent subtopic relevance model. In *CIKM*, 2011.

[32] E. Spyromitros-Xioufis, S. Papadopoulos, A. L. Ginsca, A. Popescu, Y. Kompatsiaris, and I. Vlahavas. Improving diversity in image search via supervised relevance scoring. In *ACM ICMR*, 2015.

[33] J. Wang and J. Zhu. On statistical analysis and optimization of information retrieval effectiveness metrics. In *Proceedings of ACM SIGIR conference on Research and development in information retrieval*, 2010.

[34] X.-J. Wang, L. Zhang, F. Jing, and W.-Y. Ma. Annosearch: Image auto-annotation by search. In *CVPR*, 2006.

[35] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *NIPS*, 2009.

[36] J. Weston, S. Bengio, and N. Usunier. Large scale image annotation: learning to rank with joint word-image embeddings. *JMLR*, 2010.

[37] J. Weston, S. Bengio, and N. Usunier. WSABIE: scaling up to large vocabulary image annotation. In *IJCAI*, 2011.

[38] L. A. Wolsey. *Integer programming*. 1998.

[39] H. Yu, I. Ko, Y. Kim, S. Hwang, and W.-S. Han. Exact indexing for support vector machines. In *SIGMOD*, 2011.

[40] H.-F. Yu, P. Jain, P. Kar, and I. Dhillon. Large-scale multi-label learning with missing labels. In *ICML*, 2014.