Laplacian Co-hashing of Terms and Documents

Dell Zhang[1], Jun Wang[2], Deng Cai[3], and Jinsong Lu[1]

[1] School of Business, Economics and Informatics
Birkbeck, University of London
Malet Street, London WC1E 7HX, UK
dell.z@ieee.org, jingsong.lu@gmail.com

[2] Department of Computer Science
University College London
Gower Street, London WC1E 6BT, UK
jun.wang@cs.ucl.ac.uk

[3] State Key Lab of CAD&CG, College of Computer Science
Zhejiang University
100 Zijinggang Road, 310058, China
dengcai@gmail.com

Abstract. A promising way to accelerate similarity search is *semantic hashing* which designs compact binary codes for a large number of documents so that semantically similar documents are mapped to similar codes within a short Hamming distance. In this paper, we introduce the novel problem of *co-hashing* where both documents and terms are hashed simultaneously according to their semantic similarities. Furthermore, we propose a novel algorithm Laplacian Co-Hashing (LCH) to solve this problem which directly optimises the Hamming distance.

1 Introduction

The technique of *similarity search* (aka *nearest neighbour search*) is at the core of Information Retrieval (IR) [1]. A promising way to accelerate similarity search is *semantic hashing* [2] which designs compact binary codes for a large number of documents so that semantically similar documents are mapped to similar codes (within a short Hamming distance). Existing methods for semantic hashing include stacked Restricted Boltzmann Machine (RBM) [2] and Spectral Hashing (SpH) [3] etc. In this paper, we introduce the novel problem of *co-hashing* where both documents and terms are hashed simultaneously according to their semantic similarities. This problem is different from all existing work that is about hashing one type of items only, i.e., either documents or terms. The co-hashing technique can not only provide a high hashing quality by exploiting the duality of document hashing and term hashing, but also facilitate applications that require documents and terms to be matched in the same semantic space.

Given a document collection that contains m terms and n documents, it can be represented by a $m \times n$ term-document matrix X whose rows correspond to terms and columns to documents. A non-zero entry in this matrix, say X_{ij}, indicates the presence of term i in document j, and its value reflects the strength of their association (e.g., using the TF×IDF weighting scheme [1]). Suppose that the desired length of code is l bits. We use $\mathbf{a}_i \in \{-1, +1\}^l$ and $\mathbf{b}_j \in \{-1, +1\}^l$ to represent the binary codes for term i and document j respectively, where the p-th element of $\mathbf{a}_i$ or $\mathbf{b}_j$ is $+1$ if the p-th bit of code is on or -1 otherwise. Let $A = [\mathbf{a}_1, \ldots, \mathbf{a}_m]^T$ and $B = [\mathbf{b}_1, \ldots, \mathbf{b}_n]^T$. So the problem of co-hashing can be formally defined as finding the compact binary code matrices A and B that best preserve the semantic similarity structure of a given term-document matrix X.

2 Approach

The well-known technique Latent Semantic Indexing (LSI) [1] maps both documents and terms to l-dimensional real-valued vectors in the same semantic space through truncated Singular Value Decomposition (SVD): $X \approx U_l \Sigma_l V_l^T$, where the row vectors of U_l and V_l provide the l-dimensional representation for terms and documents respectively. Therefore one simple method for co-hashing is to perform LSI first, and then binarize U_l and V_l via thresholding to get A and B. This method is named binarized-LSI in [2]. However, it can be shown that the binary codes $\mathbf{a}_1, \ldots, \mathbf{a}_m$ and $\mathbf{b}_1, \ldots, \mathbf{b}_n$ obtained in this way optimises the objective function $\sum_i \sum_j \left(\mathbf{a}_i^T \Sigma_l \mathbf{b}_j - X_{ij}\right)^2$, which is not directly related to our goal of retaining the semantic similarity structure in the Hamming space.

We propose a novel algorithm **Laplacian Co-Hashing (LCH)** which directly optimises the Hamming distance. Let's consider the document collection as an undirected *bipartite* graph [4]: the first m vertices represent the terms while the remaining n vertices represent the documents; an edge (i, j) exists if term i occurs in document j and it is weighted by X_{ij}; there are no edges between terms or between documents. The adjacency matrix of this graph can be written as $W = \begin{bmatrix} 0 & X \\ X^T & 0 \end{bmatrix}$, and its degree matrix is given by $D = \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix}$ where D_1 and D_2 are diagonal matrices such that $D_1(i, i) = \sum_j X_{ij}$ and $D_2(j, j) = \sum_i X_{ij}$. The Hamming distance between $\mathbf{a}_i$ and $\mathbf{b}_j$, is given by the number of bits that are different between them, which can be calculated as $\frac{1}{4}\|\mathbf{a}_i - \mathbf{b}_j\|^2$. To meet the similarity-preserving criterion, we seek to minimise the weighted average Hamming distance (as in [3]) $\frac{1}{4}\sum_i \sum_j W_{ij}\|\mathbf{a}_i - \mathbf{b}_j\|^2$ because it incurs a heavy penalty if a pair of strongly associated term and document are mapped far apart in the Hamming space. The above objective function can be rewritten in matrix form as $\frac{1}{4}\operatorname{Tr}(Z^T L Z)$, where $Z = \begin{bmatrix} A \\ B \end{bmatrix}$ and $L = D - W = \begin{bmatrix} D_1 & -X \\ -X^T & D_2 \end{bmatrix}$ is the *graph Laplacian* [4]. We found the above objective function actually proportional to that of a well-known manifold learning algorithm, Laplacian Eigenmap (LapEig) [5], except that LapEig does not have the constraints $\mathbf{a}_i \in \{-1, +1\}^l$ and $\mathbf{b}_i \in \{-1, +1\}^l$. So, if we relax the discreteness condition but just keep the similarity-preserving requirement, we can get the optimal l-dimensional real-valued vectors $\tilde{\mathbf{a}}_i \in \mathbb{R}^l$ and $\tilde{\mathbf{b}}_j \in \mathbb{R}^l$ by solving the LapEig problem:

$$\arg\min_{\tilde{Z}} \operatorname{Tr}(\tilde{Z}^T L \tilde{Z}) \quad \text{subject to } \tilde{Z}^T D \tilde{Z} = I \text{ and } \tilde{Z}^T D 1 = 0$$

where $\mathrm{Tr}(\tilde{Z}^T L \tilde{Z})$ gives the real relaxation of the weighted average Hamming distance, and the two constraints prevent collapse onto a subspace of dimension less than l. The solution of this optimisation problem is given by $\tilde{Z} = [\mathbf{z}_1, \dots, \mathbf{z}_l]$ whose columns are the l eigenvectors corresponding to the smallest eigenvalues (except 0) of the generalized eigenvalue problem $L\mathbf{z} = \lambda D\mathbf{z}$. Taking advantage of the special structure of L and D, we can rewrite the above equation as

$$\begin{bmatrix} D_1 & -X \\ -X^T & D_2 \end{bmatrix} \begin{bmatrix} \mathbf{s} \\ \mathbf{t} \end{bmatrix} = \lambda \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix} \begin{bmatrix} \mathbf{s} \\ \mathbf{t} \end{bmatrix}.$$

Assuming that D_1 and D_2 are non-singular, we get

$$D_1^{\frac{1}{2}}\mathbf{s} - D_1^{-\frac{1}{2}} X\mathbf{t} = \lambda D_1^{\frac{1}{2}}\mathbf{s} \quad \text{and} \quad -D_2^{-\frac{1}{2}} X^T\mathbf{s} + D_2^{\frac{1}{2}}\mathbf{t} = \lambda D_2^{\frac{1}{2}}\mathbf{t}.$$

Introducing $\mathbf{u} = D_1^{\frac{1}{2}}\mathbf{s}$ and $\mathbf{v} = D_2^{\frac{1}{2}}\mathbf{t}$, we get

$$D_1^{-\frac{1}{2}} X D_2^{-\frac{1}{2}}\mathbf{v} = (1-\lambda)\mathbf{u} \quad \text{and} \quad D_2^{-\frac{1}{2}} X^T D_1^{-\frac{1}{2}}\mathbf{u} = (1-\lambda)\mathbf{v}.$$

Letting $\check{X} = D_1^{-\frac{1}{2}} X D_2^{-\frac{1}{2}}$, we finally have

$$\check{X}\check{X}^T\mathbf{u} = (1-\lambda)^2\mathbf{u} \quad \text{and} \quad \check{X}^T\check{X}\mathbf{v} = (1-\lambda)^2\mathbf{v},$$

which means that $\mathbf{u}$ and $\mathbf{v}$ are the left and right singular vectors of $\check{X}$ respectively, while $1-\lambda$ is the corresponding singular value. Let $\check{\Sigma}_l = diag(\sigma_1, \dots, \sigma_l)$ denote the diagonal matrix that contains the largest singular values (except 1) of $\check{X}$; and let $\check{U}_l = [\mathbf{u}_1, \dots, \mathbf{u}_l]$ and $\check{V}_l = [\mathbf{v}_1, \dots, \mathbf{v}_l]$ denote respectively the matrices that consist of the corresponding l left and right singular vectors of $\check{X}$. Thus we have $\tilde{A} = D_1^{-\frac{1}{2}}\check{U}_l$ and $\tilde{B} = D_2^{-\frac{1}{2}}\check{V}_l$ as the real approximation to A and B. Moreover, given a previously unseen document $\mathbf{q}$, the real-valued l-dimensional vector $\tilde{\mathbf{q}}$ that approximates its binary code can be computed using the "fold-in" formula: $\tilde{\mathbf{q}} = \check{\Sigma}_l^{-1}\check{U}_l^T D_1^{-\frac{1}{2}}\mathbf{q}/\left(\sum_i q_i\right)$. The same method can be used for previously unseen terms as well. To convert the obtained l-dimensional real-valued term vectors or document vectors into binary codes, we threshold the p-th element of each term vector or document vector at the *median* of $D_1^{-\frac{1}{2}}\mathbf{u}_p$ or $D_2^{-\frac{1}{2}}\mathbf{v}_p$. In this way, the p-th bit will be on for half of the corpus and off for the other half. Furthermore, as the eigenvectors given by LapEig are orthogonal to each other, different bits in the generated binary codes will be uncorrelated. Therefore this thresholding method gives each distinct binary code roughly equal probability of occurring in the corpus, thus achieves the best utilization efficiency of the hash table.

3 Experiments

We have conducted experiments on two datasets, 20NG[1] and TDT2[2]. The 20NG corpus consists of 18846 documents that are evenly distributed across 20 semantic classes. The original 'bydate' split leads to 11314 (60%) documents for

[1] http://people.csail.mit.edu/jrennie/20Newsgroups/

[2] http://www.nist.gov/speech/tests/tdt/tdt98/index.htm

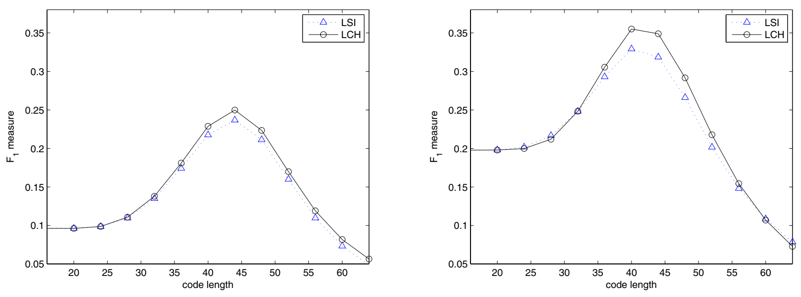

Fig. 1. The results of binarized-LSI and LCH on 20NG (left) and TDT2 (right)

training and 7532 (40%) documents for testing. The TDT2 corpus consists of 11201 documents that are classified into 96 semantic classes. In our experiments, those documents appearing in more than one class were removed, and only the largest 30 classes were kept, thus leaving us with 9394 documents in total. We randomly selected 5597 (60%) documents for training and 3797 (40%) documents for testing. Each dataset is pre-processed by stop-word removal, Porter stemming, selection of 2000 terms with the highest document frequencies (as in [2]), and TFxIDF weighting [1]. For each dataset, we use each document in the test set as a query to retrieve documents in the training set, and then compute the F_1 measure that is *micro*-averaged over all test queries [1]. To decide whether a retrieved document is relevant to the query document, we simply check whether they have the same class label (as in [2]). Figure 1 compares LCH with binarized-LSI [2] in terms of their F_1 measures with Hamming distance ≤ 16, using different length of codes. On both datasets our proposed algorithm LCH outperforms the baseline algorithm binarized-LSI consistently. We think the superior performance of LCH is attributed to its ability of directly optimising the Hamming distance.

References

1. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
2. Salakhutdinov, R., Hinton, G.: Semantic hashing. International Journal of Approximate Reasoning 50(7), 969–978 (2009)
3. Weiss, Y., Torralba, A.B., Fergus, R.: Spectral hashing. In: Proceedings of NIPS 2008, Vancouver, Canada, pp. 1753–1760 (2008)
4. Dhillon, I.S.: Co-clustering documents and words using bipartite spectral graph partitioning. In: Proceedings of KDD 2001, San Francisco, CA, USA, pp. 269–274 (2001)
5. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural Computation 15(6), 1373–1396 (2003)