Original Article
International Journal of Fuzzy Logic and Intelligent Systems
Vol. 14, No. 2, June 2014, pp. 98-104
http://dx.doi.org/10.5391/IJFIS.2014.14.2.98

ISSN(Print) 1598-2645
ISSN(Online) 2093-744X

Locality-Sensitive Hashing for Data with Categorical and Numerical Attributes Using Dual Hashing

Keon Myung Lee
Department of Computer Science, Chungbuk National University, Cheongju, Korea

ijfis

Abstract

Locality-sensitive hashing techniques have been developed to efficiently handle nearest neighbor searches and similar pair identification problems for large volumes of high-dimensional data. This study proposes a locality-sensitive hashing method that can be applied to nearest neighbor search problems for data sets containing both numerical and categorical attributes. The proposed method makes use of dual hashing functions, where one function is dedicated to numerical attributes and the other to categorical attributes. The method consists of creating indexing structures for each of the dual hashing functions, gathering and combining the candidates sets, and thoroughly examining them to determine the nearest ones. The proposed method is examined for a few synthetic data sets, and results show that it improves performance in cases of large amounts of data with both numerical and categorical attributes.

Keywords: Locality sensitive hashing, Data analysis, Search, Hashing

1. Introduction

Nearest neighbor search is a fundamental operation in data processing that determines a specified number of data objects nearest to an object specified in a query from a collection of data objects. As the amount and dimensionality of data increase, this simple operation places a considerable burden on search time and space. A related problem to which nearest neighbor search techniques are applicable is the similar pair identification problem, where all pairs of data objects similar to each other need to be found. Conventional tree-based indexing techniques, such as kd-tree and spatial tree, work well to guarantee exact solutions for low-dimensional data, but break down and almost deteriorate to an exhaustive search in cases involving high-dimensional data.

As an enabling technique for nearest neighbor search and similar pair identification for large amounts of high-dimensional data, locality-sensitive hashing (LSH) techniques use hashing to conduct rapid searches at the cost of the exactness of the results. LSH techniques create indexing structures, for which they extract short signatures for data objects using hashing functions and maintain the data objects in buckets according to their signatures. Since there is no guarantee that all similar objects are mapped to the same bucket, LSH techniques are approximate. For nearest neighbor search or similar pair identification, LSH techniques first extract the signature for a given query, and find the candidates to be exampled by calling out data objects with the same signature as the query. The volume of the resulting candidates is much smaller than that of the original data set.

Received: Jun. 1, 2014
Revised : Jun. 24, 2014
Accepted: Jun. 24, 2014

Correspondence to: Keon Myung Lee
(kmlee@cbnu.ac.kr)

When measurements are recorded from different perspectives, the corresponding data comes to have as many attributes as measurements. Some attributes are numerical and others can be categorical. Attributes are sometimes specified by a set, e.g., text values in addresses, memos, and messages. Most LSH techniques have been developed for numerical data, and a few LSH techniques have been attempted for categorical data and set-valued data [1]. Nevertheless, much data in real life contain numerical and categorical attributes together. Little attention has been paid to LSH techniques to deal with data containing such mixed attributes.

This paper is concerned with an LSH technique applicable to large collections of data with both numerical and categorical attributes. The remainder of the paper is organized as follows: Section 2 contains a description of LSH and existing techniques for numerical and categorical data, respectively. Section 3 details the proposed LSH technique to incorporate dual hash functions to handle both numerical and categorical attributes. Some experimental results of the proposed method are presented in Section 4, and conclusions are offered in Section 5.

2. Related Works

2.1 Locality-Sensitive Hashing

To efficiently locate a specific data item in a collection of data, a number of indexing structures have been developed in the literature [2–8]. Most of these adopt either tree-based or hashing-based approaches. Tree-based indexing methods encode the space partitioning information into a tree structure in which the data of interest can be found by scanning down the tree. Examples of tree-based indexing methods are kd-tree, R-tree, hierarchical k-means, ball tree, spatial tree, and spill tree [5, 9]. They are efficient at handling search tasks in low-dimensional space, but perform drastically poorly as the dimensions of space increase. Hashing is an approach for directly locating data by computing its location from its value using a special function called a hash function. A hash function is a computable map to project a large data set, called keys, to a smaller index set of a fixed length. Each index of the index set has its own bucket to which corresponding keys are mapped. When two or more data objects fall into a bucket together, they are said to collide. Since the index set is usually much smaller than the data set, buckets come to have multiple colliding data [1].

In nearest neighbor search and similar pair identification, it is desirable to find data similar to the given data rather than to locate it. LSH is an approximate search technique for data items similar to the one being queried. The query uses special hash functions that map similar data objects to the same bucket with a high probability, whereas conventional hash functions do not make any assumptions about the similarity or homogeneity in buckets.

The notion of locality-sensitive hashing was first introduced by Gionis et al. [10], Indyk et al. [11], who defined locality-sensitive hash functions as follows:

A family of functions $\mathcal{H} = \{h : S \to U\}$ is an LSH family when for any two points $p, q \in S$, for any function h from $\mathcal{H}$, the following conditions hold:

- if $d(p, q) \leq r_1$, then $Pr_{\mathcal{H}}(h(p) = h(q)) \geq P_1$.
- if $d(p, q) \geq r_2$, then $Pr_{\mathcal{H}}(h(p) = h(q)) \leq P_2$.

Here, $d(p, q)$ denotes the distance between p and q, $Pr_{\mathcal{H}}(h(p) = h(q))$ indicates the probability of $h(p) = h(q)$, r_1 and r_2 are constants for distances ($r_1 < r_2$), and P_1 and P_2 are constants for probabilities ($P_1 > P_2$). A family $\mathcal{H}$ of functions satisfying the above conditions is called (r_1, r_2, P_1, P_2)-sensitive.

Most LSH techniques encode data objects into short binary strings that play the role of signatures. Data objects with the same binary strings are maintained in the same buckets. When a query is made, it is also encoded into a binary string using the same LSH technique. Following this, data objects from the bucket with the binary string are examined according to a nearest neighbor search because the bucket might contain data objects similar to the one being queried.

2.2 LSH for Numerical Data

Most LSH techniques make use of multiple binary hash functions, each of which produces a bit value such that all bit values are subsequently concatenated into a binary string. The binary strings play the role of bucket identifiers for corresponding data objects. Many LSH techniques have been developed for data with only numerical attributes [12–18]. The following describes a few of them.

LSH hashing for binary codes by Andoni and Indyk [12], Datar et al [13] is a method that uses several hyperplanes specified randomly selected projection vectors to define hyperplanes. Each hyperplane plays the role of a hash function that produces a single bit: 1 for one side of the hyperplane and 0 for the other side.

Boost similarity-sensitive hashing [19] uses a supervised method that first samples a subset of data to determine similar and dissimilar pairs among them, and regards similar pairs as

positive examples and dissimilar pairs as negative ones. Following this, it learns weak classifiers to separate positives from negatives using the AdaBoost algorithm [20].

The restricted Boltzmann machine (RBM) is a kind of Markov random field that has a layered network to allow connections only between the visible layer and the hidden layer, where the nodes of the network are stochastic binary units. RBM can learn to maximize the probability of reconstructing the original input at the visible layer by activating the hidden layer using a contrastive divergence sampling-based update rule. The RBM-based LSH method [16] uses a stacked RBM with multiple layers, where the upper layers have a gradually decreasing number of nodes. The outputs of the topmost layer are the binary hash codes for the data fed into the bottom-most layer as input.

The spectral hashing method [18, 21] first finds the principal axes of the data set and adjusts the data along these axes. It then chooses the k smallest one-dimensional eigenfunctions to produce binary codes for data by thresholding the eigenfunction values for data at level zero.

In semi-supervised hashing [17, 22], both labeled and unlabeled data pairs are used to determine projection vectors to minimize the empirical error in the labeled data while maximizing the entropy of the generated hash bits over the unlabeled data. The projection vectors are obtained by eigendecomposing a $K \times K$ matrix, where K is the number of hash functions.

In unsupervised sequential projection learning for hashing [17], hash functions are sequentially learned as pseudo-labels are generated at each iteration. Subsequent hash functions are determined so that they correct errors made by the preceding hash functions. The pseudo-labels are assigned to pairs of closed data points residing on the opposite side of the hyperplane and pairs of far data points on the same side of the hyperplane. Once pseudo-labels are assigned, the method works in a similar manner to semi-supervised hashing [22].

In density-sensitive hashing [15], hash functions are determined by taking into account the distribution of the data set. The LSH method first applies a k-means algorithm to the data set to generate small groups and determines pairs of adjacent groups. It then finds median planes to separate adjacent groups and evaluates them according to their entropy. It chooses as hash functions the top-ranked median planes in order of descending entropy score.

2.3 LSH for Categorical Data

In LSH for categorical data [1], the similarity matrix for categorical values is first determined for each categorical attribute using a data-driven distance [23]. A hierarchical clustering is then carried out for categorical values with respect to the similarity matrix for them. Following this, clusters for categorical values are selected at an appropriate level of the dendrogram of the hierarchical cluster. Each cluster now contains a set of categorical values and is assigned a binary code. A hash function is defined for each categorical attribute that produces a binary code corresponding to the value of the categorical attribute. When there are multiple attributes in data, an LSH function is defined by combining the hash functions for the categorical attributes. The LHS function produces a binary string for categorical values and associates a bucket with a binary string in the domain of the LSH function.

3. Locality-sensitive Hashing for Data with Numerical and Categorical Attributes

There is no method of assigning data with categorical attributes in a coordinate system because there is no inherent ordering relationship among categorical values. When numerical attributes occur together with categorical attributes, data objects cannot be projected into a coordinate system. LSH techniques for numerical data basically partition the (Euclidean) data space into subspaces, each of which corresponds to a bucket of a hash function. It is not easy to have available a distance measure for data having both numerical and categorical attributes. Hence, it is better to consider numerical attributes and categorical attributes separately. The similarity between a query and data object is regarded as high when both the similarity of numerical attributes and that of categorical attributes are simultaneously high. When we collect the candidate data into a query, it is effective to take the intersection of the data objects with numerical attributes similar those of the query, as well as that of data objects with categorical attributes similar to those of the query. We thus propose an LSH technique that takes this approach.

3.1 The Proposed Dual Hashing Structure

To support LSH for data with numerical and categorical attributes, our proposed method uses two ways of hashing: one for numerical attributes and the other for categorical attributes. For convenience of description, we use the following notations: $A = \{A_{N_1}, \ldots, A_{N_p}, A_{C_1}, \ldots, A_{C_q}\}$ denotes the set of numer-

ical attributes with p numerical attributes $A_{N_1}, \ldots, A_{N_p}$, and q categorical attributes $A_{C_1}, \ldots, A_{C_q}$. $\mathcal{H}_N(A_{N_1}, \ldots, A_{N_p}; Q)$ indicates the hashing result for numerical attributes of a query Q using hashing function $\mathcal{H}_N$, and $\mathcal{H}_C(A_{C_1}, \ldots, A_{C_q}; Q)$ is the hashing result for the categorical attributes of a query Q using hashing function $\mathcal{H}_C$. $B_{N_1}, \ldots, B_{N_m}$ and $B_{C_1}, \ldots, B_{C_n}$ are buckets for the corresponding hashing values of hash functions $\mathcal{H}_N$ and $\mathcal{H}_C$, respectively. Let $D = \{D_1, \ldots, D_s\}$ be the given data set where $D_i = (d^i_{N_1}, \ldots, d^i_{N_p}, d^i_{C_1}, \ldots, d^i_{C_q})$ is the i-th data item.

Figure 1 shows the proposed dual hashing structure for data with both numerical and categorical attributes. For the given data set D, the proposed method maps each data object D_i into a numerical bucket $B_{N_{\mathcal{H}}(D_i)}$ and a categorical bucket $B_{C_{\mathcal{H}}(D_i)}$. The buckets maintain only the IDs of data objects belonging to them in order to not store duplicated copies of data objects.

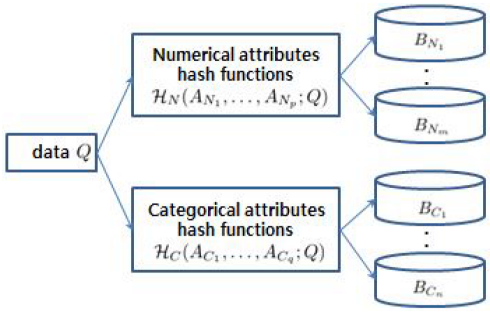


Figure 1. Dual hashing for numerical and categorical attributes.

When a query Q is given, its numerical hash code h_N and categorical hash code h_C are computed as follows:

$$h_N = \mathcal{H}_N(A_{N_1}, \ldots, A_{N_p}; Q) \tag{1}$$

$$h_C = \mathcal{H}_C(A_{C_1}, \ldots, A_{C_q}; Q) \tag{2}$$

The candidate data set D_Q similar to query Q is the intersection of the corresponding buckets $B_{N_{\mathcal{H}}}$ and $B_{C_{\mathcal{H}}}$.

$$D_Q = B_{N_{\mathcal{H}}} \cap B_{C_{\mathcal{H}}} \tag{3}$$

3.2 LSH for Numerical Attributes

To get the specified number k of nearest neighbors to the query Q, the numerical similarity $S_N(D_i, Q)$ and the categorical similarity $S_C(D_i, Q)$ are computed for each data item $D_i \in D_Q$. When $S_N(D_i, Q)$ is computed, only the numerical attribute values $(D^i_{N_1}, \ldots, D^i_{N_p})$ of D_i and the corresponding numerical attribute values of query Q are considered. Most LSH techniques for numerical data assume that similarities are computed through Euclidean distance after the standardization of data to have mean 0 and standard deviation 1.

3.3 LSH for Categorical Attributes

A typical distance measure $d_k(a, b)$ for categorical values a and b is defined as $d_k(a, b) = 0$ if $a = b$; otherwise, $d_k(a, b) = 1$. That is, it outputs 1 as the distance when they are different, and 0 when the two values are the same. The corresponding similarity measure $S^T_k(a, b)$ is defined as follows:

$$S^T_k(a, b) = \begin{cases} 1 & \text{if } a = b \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Hence, the measure cannot have any distance information when the compared categorical values are different. To handle this problem, data-driven distance measures have been studied for data with categorical attributes [23]. They make use of attribute value distributions in a given data set and determine the distances between attribute values, i.e., between categorical values. Boriah et al. [23] have surveyed data-driven similarity measures on categorical values. Three measures among them are presented in the following, where $S_j(a, b)$ denotes the similarity of categorical values a and b for the j-th attribute A_{C_j}, $f_k(a)$ is the number of times attribute A_{C_k} takes the value a in data set D, and $p_k(a)$ is the sample probability of A_{C_k} to take a, i.e., $p_k(a) = f_k(a)/N$, where N is the number of data in D. $p^2_k(a)$ is another probability estimate of A_{C_k} to take a, defined as $p^2_k = f_k(a)(f_k(a) - 1)/(N(N - 1))$.

(IOF measure)

$$S^I_k(a, b) = \begin{cases} 1 & \text{if } a = b \\ \frac{1}{1+\log f_k(a) \log f_k(a)} & \text{otherwise} \end{cases} \tag{5}$$

(OF measure)

$$S^O_k(a, b) = \begin{cases} 1 & \text{if } a = b \\ \frac{1}{1+\log \frac{N}{f_k(a)} \log \frac{N}{f_k(b)}} & \text{otherwise} \end{cases} \tag{6}$$

(Goodall measure)

$$S^G_k(a, b) = \begin{cases} 1 - \sum_q p^2_k(q) & \text{if } a = b \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

When there are multiple attributes $A_{C_1}, A_{C_2}, \ldots, A_{C_q}$, the similarity $S(D_i, D_j)$ is computed by the weighted sum of the similarity of each attribute. Here, ω_k is the weighting factor for

attribute A_{C_k} and $av_k(D_i)$ is the k-th attribute value of the i-th data item D_i.

$$S(D_i, D_j) = \sum_{k=1}^{q} \omega_k S_k(av_k(D_i), av_k(D_j)) \quad (8)$$

The similarities are combined into a single value $S(D_i, Q)$ using a weighted sum, where $w \in (0, 1)$.

$$S(D_i, Q) = wS_N(D_i, Q) + (1 - w)S_C(D_i, Q) \quad (9)$$

4. Experiments

To test the performance of the proposed method, we conducted a few experiments under the following conditions: for LSH with numerical attributes, we used density-sensitive hashing, which explores the geometric structure of data to generate hash codes. For LSH with categorical attributes, we used data-driven categorical LSH [1], which makes use of the data-driven similarity measure for categorical values.

We conducted experiments for two synthetically generated data sets. Each synthetic data set consists of 1,000,000 data objects with 10 numerical attributes and five categorical attributes, each of which has 10 categorical values. The first data set $\mathcal{D}_1$ was generated from uniformly distributed spaces in which numerical attributes were drawn from the space $[0, 10]^{10}$, and the categorical attributes had uniform probabilistic distributions over the 10 categorical values. The second data set $\mathcal{D}_2$ was generated to contain 40 clusters. In each of these, the cluster mean vectors were randomly selected and the corresponding covariance matrices were set to $diag(3, 3, \ldots, 3)$. Each categorical attribute was set to follow a multinomial distribution, the probabilities of which are selected from a uniform Dirichlet distribution $Dir(0.1, 0.1, \ldots, 0.1)$. A total of 25,0000 data objects were generated for each cluster.

For numerical attributes, 10-bit LSH codes were generated using density-sensitive hashing. For categorical attributes, the categorical values were encoded into three disjoint groups, and 3^5 indices were created altogether. For both data sets $\mathcal{D}_1$ and $\mathcal{D}_2$, the 30 queries randomly generated were handled for candidates suggested by the proposed method, and the five nearest neighbors were searched for. The evaluation was carried out using the ground truth created through an exhaustive search of the entire data for the data being queried. When the candidate subset was selected using the LSH method for numerical attributes, the data set having the same hash code as that of the query was chosen without considering neighboring buckets. In the hash codes of LSH for categorical attributes, the Hamming distance between hash codes does not contain any neighborhood information. Table 1 shows the experimental results in terms of recall, where *Num. LSH* indicates the LSH method that considers only numerical attributes, *Cat. LSH* is the LSH method that takes into account only categorical attributes, and *Mixed LSH* denotes the proposed dual hashing technique that uses both LSH methods together.

Table 1. Experimental results of the proposed method

Data set	*Num. LSH*	*Cat. LSH*	*Mixed LSH*
$\mathcal{D}_1$	81.3	74.9	90.7
$\mathcal{D}_2$	82.4	75.4	92.8

LHS, locality-sensitive hashing

5. Conclusions

For big data applications, nearest neighbor search and similar pair identification can be burdensome tasks even though they are rather fundamental operations. Various LSH techniques have been developed to handle such tasks in an efficient but approximate manner. Most LSH techniques are applicable to only numerical data, and little attention has thus far been paid to how LSH can be engineered for data sets containing numerical and categorical attributes together. We proposed in this paper a method to combine numerical and categorical LSH techniques to handle this problem. From the experiments, we concluded that the proposed approach provides improved performance, as we had expected.

Conflict of Interest

No potential conflict of interest relevant to this article was reported.

Acknowledgments

This work was supported by a research grant from Chungbuk National University in 2012.

References

[1] K. M. Lee, "Locality-Sensitive Hashing Techniques for Nearest Neighbor Search," *Int. Journal of Fuzzy Logic and*

Intelligent Systems, Vol.12, No.4, pp.300-307, Dec. 2012. http://dx.doi.org/10.5391/IJFIS.2012.12.4.300

[2] J. L. Bentley, "Multidimensional Binary Search Trees used for Associative Searching," *Communications of the ACM*, vol. 18, no. 9, pp. 509-517, Sep. 1975. http://dx.doi.org/10.1145/361002.361007

[3] T. H. Cormen, C. E. Leiserson, R. L. Rivest, *Introduction to Algorithms,* 2nd ed., Cambridge, MA: MIT Press, 2001.

[4] D. W. Kim, K. H. Lee, "A fuzzy clustering algorithm for clustering categorical data," *Journal of The Korean Institute of Intelligent Systems,* vol.13, no.6, pp.661-666, Dec. 2003.

[5] S. M. Omohundro, *Five balltree construction algorithms*, , International Computer Science Institute Technical Report, 1989. Available ftp://ftp.icsi.berkeley.edu/pub/techreports/1989/tr-89-063.pdf

[6] S. Pandey, A. Broder, and F. Chierichetti, "Nearest-Neighbor Caching for Content-Match Applications," in *Proceedings of the 18th International Conference on World Wide Web*, Madrid, Spain, April 20-24, 2009, pp. 441-450. http://dx.doi.org/10.1145/1526709.1526769

[7] M. Potthast and B. Stein, "New Issues in Near-Duplicate Detection," in *Data Analysis, Machine Learning and Applications,* C. Preisach, H. Burkhardt, L. Schmidt-Thieme, and R. Decker, Eds. Heidelberg, Germany: Springer Berlin, 2008, pp. 601-609. http://dx.doi.org/10.1007/978-3-540-78246-9_71

[8] J. K. Uhlmann, "Satisfying general proximity/similarity queries with metric trees," *Information Processing Letters,* vol.4, no. 4, pp.175–179, Nov. 1991. http://dx.doi.org/10.1016/0020-0190(91)90074-R

[9] D. Nister and H. Stewenius, "Scalable Recognition with a Vocabulary Tree," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, New York, NY, June 17-22, 2006, pp. 2161-2168. http://dx.doi.org/10.1109/cvpr.2006.264

[10] A. Gionis, P. Indyk, and R. Motwani, "Similarity Search in High Dimensions via Hashing," in *Proceedings of the 25th International Conference on Very Large Data Bases*, Edinburgh, UK, September 7-10, 1999, pp. 518-529.

[11] P. Indyk and R. Motwani, "Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality," in *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, Dallas, TX, May 24-26, 1998, pp. 604-613. http://dx.doi.org/10.1145/276698.276876

[12] A. Andoni and P. Indyk, "Near-optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions," *Communications of the ACM*, vol. 51, no. 1, pp. 117-122, Jan. 2008. http://dx.doi.org/10.1145/1327452.1327494

[13] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive Hashing Scheme based on p-stable Distribution," in *Proceedings of the 20th Annual Symposium on Computational Geometry*, Brooklyn, NY, June 8-11, 2004, pp. 253-262. http://dx.doi.org/10.1145/997817.997857

[14] K. M. Lee and K.M. Lee, "A Locality Sensitive Hashing Technique for Categorical Data," *Applied Mechanics and Materials*, vol. 241-244, pp. 3159-3164, Dec. 2012. http://dx.doi.org/10.4028/www.scientific.net/AMM.241-244.3159

[15] Y. Lin, D. Cai, "Density Sensitive Hashing," Submitted on May 14, 2012. Available http://arxivorg/abs/12052930

[16] R. R. Salakhutdinov and G.E. Hinton, "Semantic hashing," *International Journal of Approximate Reasoning*, vol. 50, no. 7, pp. 969-978, Jul. 2009. http://dx.doi.org/10.1016/j.ijar.2008.11.006

[17] J. Wang, S. Kumar, and S.-F. Chang, "Sequential Projection Learning for Hashing with Compact Codes," in *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, June 21-24, 2010.

[18] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Proceedings of the 22nd Annual Conference on Neural Information Processing Systems*, Vancouver, Canada, December 8-10, 2008, pp. 1753-1760.

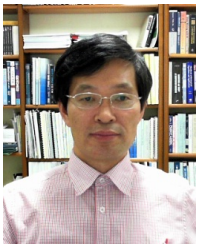
[19] G. Shakhnarovich, P. Viola, and T. Darrell, "Fast Pose Estimation with Parameter Sensitive Hashing," in *Proceedings of the 9th IEEE International Conference on Computer Vision*, Nice, France, October 13-16, 2003, pp. 750-757. http://dx.doi.org/10.1109/ICCV.2003.1238424

[20] R. E. Schapire, "The Boosting Approach to Machine Learning : An Overview," in *Nonlinear Estimation and Classification*, D. Denison, M. Hansen, C. Holmes, B. Mallick, and B. Yu, Eds. New York, NY: Springer, 2003, pp. 149-171. http://dx.doi.org/10.1007/978-0-387-21579-2_9

[21] U. von Luxburg, "A Tutorial on Spectral Clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395-416, Dec. 2007. http://dx.doi.org/10.1007/s11222-007-9033-z

[22] J. Wang, S. Kumar, and S.-F. Chang, "Semi-Supervised Hashing for Large Scale Search," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 12, pp. 2393-2406, Dec. 2012. http://dx.doi.org/10.1109/TPAMI.2012.48

[23] S. Boriah, V. Chandola, V. Kumar, "Similarity Measures for Categorical Data: A Comparative Evaluation," in *Proceedings of the 8th SIAM International Conference on Data Mining*, Atlanta, GA, April 24-26, 2008, pp. 243-254.



Keon Myung Lee is a professor at Department of Computer Science, Chungbuk National University, Korea. He received his BS, MS, and Ph.D. degrees in computer science from KAIST, Korea and was a Post-doc fellow in INSA de Lyon, France. He was a visiting professor in University of Colorado at Denver and a visiting scholar in Indiana University, USA. His principal research interests are in data mining, machine learning, soft computing, big data processing, and intelligent service systems.