Expert Systems With Applications 62 (2016) 202–211

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Hashing-based clustering in high dimensional data

Juan Zamora[a], Marcelo Mendoza[b,*], Héctor Allende[a]

[a] Department of Informatics, Universidad Técnica Federico Santa María, España Avenue 1680, Valparaíso, P.O. 2340000, Chile
[b] Department of Informatics, Universidad Técnica Federico Santa María, Vicuña Mackenna Avenue 3939, Santiago, P.O. 8940000, Chile

ARTICLE INFO

Article history:
Received 13 April 2015
Revised 6 June 2016
Accepted 7 June 2016
Available online 16 June 2016

Keywords:
Locality sensitive hashing
High dimensional clustering
Min-wise hashing
Random hyperplanes

ABSTRACT

Clustering is one of the most important techniques for the design of intelligent systems, and it has been incorporated into a large number of real applications. However, classical clustering algorithms cannot process high-dimensional data, such as text, in a reasonable amount of time. To address this problem, we use techniques based on locality-sensitive hashing (LSH), which was originally designed as an efficient means of solving the near-neighbor search problem for high-dimensional data. We propose the use of two LSH strategies to group high-dimensional data: MinHash, which enables Jaccard similarity approximations, and SimHash, which approximates cosine similarity. Instead of creating a computational costly data structure for responding to queries from near neighbors, we use a low-dimensional Hamming embedding to approximate a pairwise similarity matrix using a single-pass procedure. This procedure does not require data storage. It requires only the maintenance of a low-dimensional embedding. Then, the clustering solution is found by applying the bisection method to the similarity matrix. In addition to the above, we propose an improvement to LSH that is beneficial for its use on high-dimensional data. This improvement introduces a penalty on the Hamming distance, which is used in conjunction with SimHash, thereby improving the cosine similarity approximation. Experimental results indicate that our proposal yields a solution that is very close to the one found by applying the bisection method to a matrix with complete information, with better running times and a lower use of memory.

1. Introduction

Handling large volumes of data is one of the tasks that pose the greatest challenges in the information age. The need to record, characterize and organize data is vital to the design of intelligent information systems. Information systems should be able to summarize data repositories of highly various natures to provide relevant and timely information to their users.

Clustering is a common task in the design of information systems because it allows similar objects to be organized into groups. Clustering is beneficial for the performance of various tasks relevant to the design of intelligent information systems, such as the cataloging, indexing, search, retrieval, characterization and summarization of data.

Data can be diverse in format and content, can originate from heterogeneous sources, and can arrive at regular intervals or in large batches over short periods of time; they may be incomplete, imperfect or inconsistent. Because of this, there is no single clustering algorithm that can cope with all of the above challenges. Almost fifty years after the publication of the most famous of all clustering algorithms k-means, the problem is far from solved.

In this paper, we focus on addressing the problem of clustering in high-dimensional data. High dimensionality is a result of the difficulty of producing sufficient descriptions of the content of data using only a few descriptors. Typically, high dimensionality is common in text data because the content descriptors are words, which are abundant in quantity, meanings and uses.

Text is organized into logical units known as documents. The storage and retrieval of information in and from documents require clustering algorithms that can effectively group similar documents across high-dimensional feature spaces. This task is critical to the design of intelligent systems for the storage and retrieval of news, patents, websites, tweets, short messages or scientific articles, among other applications (Leskovec, Rajaraman, & Ullman, 2014). Without clustering in these systems, it would not be possible to organize documents into collections; such organization facilitates key tasks at the levels of storage (vertical search (Zhou, Cummins, Lalmas, & Joemon, 2013)), recovery (collection selection (Puppin, Silvestri, Perego, & Baeza-Yates, 2010)) and synthesis (multi-document summarization (Radev, Jing, Stys, & Tam, 2004)).

* Corresponding author.
E-mail addresses: juan.zamora@usm.cl (J. Zamora), mmendoza@inf.utfsm.cl, marcelo.mendoza@usm.cl (M. Mendoza), hallende@inf.utfsm.cl (H. Allende).

http://dx.doi.org/10.1016/j.eswa.2016.06.008
0957-4174/

The need to organize documents from the Web has driven the design of efficient storage strategies for the management of high-dimensional data. Broder (1997) showed that by quantizing the bit vector of each document into short segments, known as shingles, and applying random sampling to each shingle, it is possible to detect near-duplicates. The idea is that the number of matching shingles between a pair of documents enables the detection of near-duplicates, a concept called document resemblance. This strategy enabled the design of a process for the collection of documents from the Web (crawling) that is able to eliminate near-duplicates, thereby avoiding redundancy. This strategy is called syntactic clustering (Broder, Glassman, Manasse, & Zweig, 1997).

Almost simultaneously, Indyk and Motwani (1998) introduced strategies based on hashing functions that allow the indexing of similar documents to adjacent addresses in the main memory. These strategies, known as locality-sensitive hashing (LSH), apply several hashing functions to the vector terms of each document, which are usually binary (bit vectors). Thus, each document is assigned a signature corresponding to the sequence of bucket IDs in which the document is stored. Because the number of matching buckets between two documents approximates the Jaccard pairwise similarity, LSH is useful for searching for near neighbors in high-dimensional data. Subsequently, Gionis, Indyk, and Motwani (1999) improved the technique by making it more efficient and allowing searches of the secondary memory, which is advantageous for the management of larger collections. This technique is known as MinHash.

The ideas behind MinHash and document resemblance are similar. Using randomization (random sampling in resemblance or hashing in LSH), it is possible to approximate the pairwise similarity functions between pairs of documents in a space of lower dimensionality (fixed-length signatures or hash code bit vectors). For the task of finding high-dimensional near neighbors, an exact answer is not required; a good approximation suffices.

A modification to the MinHash strategy allowed Charikar (2002) to obtain an approximation of the cosine similarity. This modified strategy, known as SimHash, modifies the hashing function such that the hash codes indicate the orientation of each document in relation to a random hyperplane. In this case, the number of matching buckets between two documents corresponds to an approximation of the cosine of the angle between their vectors.

Both MinHash and SimHash project the original feature space to a lower-dimensional space, defined by the length of the hash code vector. The resulting space corresponds to a $\{0, 1\}^d$ Hamming embedding, where d is the dimensionality of the projected space (the length of the hash code vector). For the MinHash and SimHash strategies, the Hamming distances in this embedding correspond to the Jaccard and Ochini distances, respectively. Both strategies provide guarantees regarding the approximation of the similarity function.

In this article, we explore the use of MinHash and SimHash on group data of high dimensionality. Instead of storing the data in multiple hash tables in memory, which is essential for finding near neighbors, we calculate only the similarities between objects that are close with high probability, eliminating the need to store data in memory. This allows us to reduce the memory required by the algorithm. The distances are stored in an index of neighbors using adjacency lists, which allows us to manage the closest pairs in memory and to neglect distant pairs or isolated data. This is achieved by adjusting the LSH parameters to limit the number of false positives that produce matches in buckets. The adjacency lists are updated using a document-at-a-time (DAAT) procedure, which reduces the cost associated with calculating the similarity matrix from $O(n^2 \cdot d)$ to $O(B \cdot n)$, where n is the number of documents in the collections, d is the dimensionality of the original space, and B is the number of non-empty LSH buckets. By applying divisive clustering (the bisection method) to the lists, clusters of documents are produced. In addition to the above, in this paper, we introduce a variant of SimHash. In this variant, the approximate calculation of the cosine similarity is modified by the introduction of a penalty factor proportional to the decimal representation of the hash code. This is because the decimal magnitude of the hash code represents the distance of the projection of an object to the corresponding random hyperplane. The proposed variant allows us to improve the approximation of the cosine similarity. To demonstrate the validity of our algorithms, we present experiments conducted on high-dimensional datasets with thousands or tens of thousands of features. The experimental results validate the algorithms in terms of the entropy and purity of the identified clusters. Our results are compared with the clusters obtained using the similarity matrix on the original space. The differences in entropy and purity vary depending on the LSH parameters and the specific LSH strategy used (MinHash or SimHash). The clusters found using the approximate method are similar to those found with the original feature space. This shows that the elimination of distant pairs or isolated data from the original dataset, which is the characteristic trait of LSH, is beneficial for the task of clustering high-dimensional data.

2. Related work

2.1. Locality sensitive hashing (LSH)

Numerous theoretical and practical advances have been made in regard to the problem of searching for near neighbors using LSH. Datar, Immorlica, Indyk, and Mirrokni (2004) have presented a new LSH strategy that is capable of projecting the original feature space to a Euclidean space using p-stable distributions. This strategy is called E2LSH. The space used by the E2LSH data structure depends on the approximation factor; as a result, this approach can be impractical even for small datasets. However, Panigrahy (2006) successfully improved the technique by building a spatial structure independent of the approximation factor used by E2LSH. Based on this structure, Andoni and Indyk (2008) were able to improve the query time, nearly reaching the lower bound for the problem found by Motwani, Naor, and Panigrahy (2006). A modified version of MinHash was proposed by Li and König (2011), called b-Bit Minwise Hashing, which limits the length of the hash code vectors to b-bits, thereby improving the spatial requirements of the data structure while maintaining similar near-neighbor search results. Recently, Shrivastava and Li (2014) showed that when the data are binary, MinHash often exceeds the performance of SimHash in terms of near-neighbor search approximations.

The growing interest in exploring LSH has also permeated the machine-learning community. Semantic hashing (Salakhutdinov & Hinton, 2009) shows that it is possible to learn hashing functions using deep learning. With the incorporation of a fine-tuning phase, the hash code vectors thus produced can occupy less memory space than those produced by LSH, reducing the near neighbor search times for documents. However, the produced codes project the original space to a dense space, which limits the separability of the data. Furthermore, this technique has shown a high parametric sensitivity. Weinberger, Dasgupta, Langford, Smola, and Attenberg (2009) proposed an efficient solution based on eigenvectors that generates the best binary codes for indexing. The sparsified version of spectral hashing (Shao, Wu, Ouyang, & Zhang, 2012) also offers better guarantees in terms of data separability. The primary weakness of learning-based methods is their excessive cost in terms of computation time during the data-structure construction phase. This limits their use on real datasets.

Finally, the use of LSH techniques has enabled the efficient performance of various data-mining tasks. These include outlier