Search index building for numeric vectors with O(1) memory consumption

Sukhodolsky A., Tytenko S.

Sukhodolsky A. Search index building for numeric vectors with O(1) memory consumption / Sukhodolsky A., Tytenko S. // Modern Aspects of Software Development: Proceedings of VI International Scientific and Practical Virtual Conference of Software Development Specialists, June, 24 2019. – Kyiv: Igor Sikorsky KPI, 2019. – pp. 54-60.

Introduction

Nowadays, the world of information technologies produces a large amount of information, and therefore we should find new ways for its effective storage and retrieval. For classical data types (text, numbers, dates, etc.) there is a large number of algorithms that allow us to index and search for almost linear time. For a relatively young vector data representation, there is also a set of algorithms, but in most cases it is necessary to find a balance between memory consumption, search speed and quality of the results.

Vector representation of data is a representation, where each unit of data represents by one-dimensional vector with dimension shape d. Unlike classical data types, this representation is hard to interpret, but it's suitable for comparative characteristics (euclidean distance or cosine similarity). This representation can be used in searching for semantically similar images or texts that were previously processed by a neural network or another method of data vectorization. One of the most important tasks in vector data processing is the nearest neighbor search (NNS), which main goal is the accurate searching the nearest neighbor vector by query vector in a short amount of time. The task is complicated by the fact that thousands of vectors consume a lot of memory, and the linear search of all vectors has complexity of $O(dN)$, where N is the vectors count. In order to simplify this task, it can be reformulated as "approximate NNS", and this new formulation has given a lot of space for new algorithms such as Product Quantization (PQ) [1], Locality Sensitive Hashing (LSH) [2], and many others. These algorithms make an approximated search for the nearest neighbors, respectively, the accuracy of final results is not guaranteed, but the speed of the algorithm can increase to $O(logN)$. However, initialization and building of a search index for these algorithms often require loading of all data into RAM, which is not possible in a case of large amount of data.

In this paper we have considered and proposed methods for building of search index for vector representation of data in $O(1)$ memory consumption using PQ, LSH and Welford method [3] for covariation matrix and mean values finding.

Formulation of problem

The main goal of this paper is to solve the problem of search index building for vector data with $O(1)$ memory consumption. Index building consists of two steps:

starting initialization and data indexing. The first step is required for vectors compression using PQ algorithm and for their whitening (PCA whitening [4]). The second step builds index on input data using LSH. To achieve $O(1)$ memory consumption, PQ and PCA whitening algorithms need to be adapted, since the first one uses K-Means clustering algorithm [5] for compression, and the second one is necessary to find the covariance matrix and the mean values along each vector's dimension.

Adaptation of PQ algorithm

Product Quantization algorithm uses K-Means clustering for vectors compression. To perform this, it divides each vector into M equal parts with dimension $\hat{d} = \frac{d}{M}$ After dividing, PQ cluster given sub-vectors into K (most often 2^8, because cluster indices can fit in one byte) centroids. After all preparations, vector can be compressed by the replacement of each sub-vector by the nearest centroid index, and thus vector size can be reduced to M bytes.

The classic K-Means algorithm requires a whole bunch of data, but in this case it is appropriate to use the online version [6] of the algorithm, which incrementally updates cluster's centroids by vectors stream. Cons of this approach are the lesser speed and instability of convergence.

Adaptation of PCA whitening algorithm

To receive better LSH algorithm performance, input vectors should be whitened. This operation allows LSH to divide vectors by a set of hyperplanes into balanced binary tree. For PCA whitening algorithm it is necessary to find the covariance matrix Σ and mean values μ for each dimension. Welford's method allows finding the covariance matrix and the mean values with $O(1)$ memory consumption.

$$\Sigma_{\mathrm{n}} = \frac{n-2}{n-1}\Sigma_{n-1} + \frac{1}{n}(x - \mu_{n-1})(x - \mu_{n-1})^T \tag{1}$$

$$\mu_n = \frac{x - \mu_{n-1}}{n-1} \tag{2}$$

To whiten the vectors, the PCA decomposition of the covariance matrix $\Sigma = U\Lambda V^T$ needs to be found.

$$\hat{x} = \Lambda^{-1/2}\mathrm{U}^{\mathrm{T}}(\mathrm{x} - \mu) \tag{3}$$

As a result, the vectors have the same distribution on every dimension and form the "bubble" on the hyperplane. Visualization of the whitening process is depicted in Figure. 1.

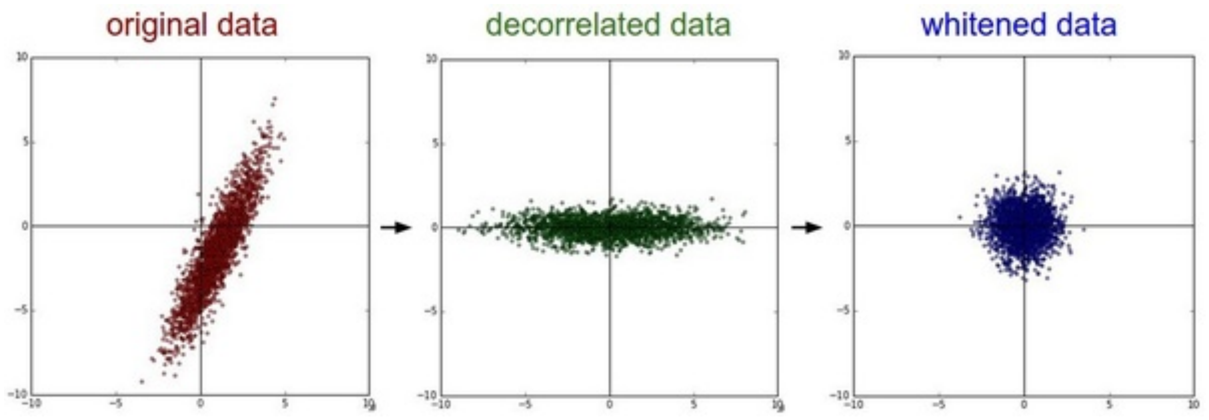

Figure 1 — Data whitening process

Data indexing using LSH

The index building is carried out using the LSH algorithm. This algorithm defines $\hat{N}$ hash functions that produce 0 or 1 values.

$$h_i(x) = \begin{cases} 1, & if\ W_i x > 0 \\ 0, & otherwise \end{cases}, \tag{4}$$

where W_i is the normal vector of hyperplane i, and the sign of the expression $W_i x$ defines the position of the vector x relatively to the hyperplane i. Since the vectors after whitening process have a “bubble” distribution, the W_i vectors can be defined as identity matrix W with shape $N \times N$.

The combined hash function is defined as an array of N hash functions.

$$h(x) = \{h_1(x),\ h_2(x),\ \cdots,\ h_N(x)\} \tag{5}$$

The resulting hashes can be compared with each other using the Hamming distance [7], which is defined as $xor(h(x_1),\ \ h(x_2))$. The resulting value of Hamming distance correlates with the cosine similarity between input vectors, and therefore this fact can be used for an approximated search.

Since the values of the combined hash functions can collide with each other, nearest vector finds by linear search among vectors with same combined hash function result. Thus, the search speed in the best case decreases to $O(\frac{N}{2^{\hat{N}}})$.

Conclusion

In this paper, several methods have considered and proposed for search index initialization and constructing for vector data types with $O(1)$ memory consumption. This approach solves the problem by finding the clusters’ centroids in PQ compression step using online K-Means algorithm and finding the covariance matrix and mean values using Welford method. The disadvantages of this approach are the low quality of the found PQ centroids and low performance of initialization step.

References

1. Jegou H. Product quantization for nearest neighbor search / H. Jegou, M. Douze, C. Schmid // IEEE transactions on pattern analysis and machine intelligence. — 2010. — Vol. 33, No. 1. — P. 117–128.
2. Kulis B. Kernelized locality-sensitive hashing for scalable image search. / B. Kulis, K. Grauman. — 2009.
3. Welford B. Note on a method for calculating corrected sums of squares and products / B. Welford // Technometrics. — 1962. — Vol. 4, No. 3. — P. 419–420.
4. Nam W. Local decorrelation for improved pedestrian detection / W. Nam, P. Dollár, J. H. Han. — 2014.
5. Hartigan J. A. Algorithm as 136: A k-means clustering algorithm / J. A. Hartigan, M. A. Wong // Journal of the Royal Statistical Society. Series C (Applied Statistics). — 1979. — Vol. 28, No. 1. — P. 100–108.
6. Liberty E. An algorithm for online k-means clustering / E. Liberty, R. Sriharsha, M. Sviridenko. — SIAM, 2016.
7. Norouzi M. Hamming distance metric learning / M. Norouzi, D. J. Fleet, R. R. Salakhutdinov. — 2012.