Balancing clusters to reduce response time variability in large scale image search

Romain Tavenard
Université de Rennes 1 / IRISA
firstname.lastname@irisa.fr

Hervé Jégou
INRIA Rennes / IRISA
firstname.lastname@inria.fr

Laurent Amsaleg
CNRS / IRISA
firstname.lastname@irisa.fr

Abstract

Many algorithms for approximate nearest neighbor search in high-dimensional spaces partition the data into clusters. At query time, for efficiency, an index selects the few (or a single) clusters nearest to the query point. Clusters are often produced by the well-known k-means approach since it has several desirable properties. On the downside, it tends to produce clusters having quite different cardinalities. Imbalanced clusters negatively impact both the variance and the expectation of query response times. This paper proposes to modify k-means centroids to produce clusters with more comparable sizes without sacrificing the desirable properties. Experiments with a large scale collection of image descriptors show that our algorithm significantly reduces the variance of response times without severely impacting the search quality.

1 Introduction

Finding the nearest neighbors of high-dimensional query points still receives a lot of research attention as this fundamental process is central to many content-based applications. Most approaches rely on some different kinds of partitioning of the data collection into clusters of descriptors. At query time, an indexing structure selects the few (or a single) clusters nearest to the query point. Each candidate cluster is probed, actual distances to its points are computed and the query result is built based on these distances.

There are various options for clustering points, the most popular being the k-means approach. Its popularity is caused by its nice properties: it is a simple algorithm, surprisingly effective and easy to implement ; it nicely deals with the true distribution of data in space by minimizing the mean square error over the clustered data collection. On the downside, it tends to produce clusters having quite different cardinalities. This, in turn, impacts the performance of the retrieval algorithm: scanning heavily filled clusters is costly as the distances to many points must be computed. In contrast, under-filled clusters are cheap to process, but they are selected less often as the query descriptor is also less likely to be associated with these less populated clusters. Overall, having imbalanced clusters impacts both the variance and the expectation of query response times. This is very detrimental to contexts in which performance is paramount, such as high-throughput settings since the true resource consumption can no more be accurately predicted by costs models. This phenomenon has an even more detrimental impact at large scale. In this case, clusters must be stored on disks and the performance severely suffer when fetching large clusters due to the large I/Os. Furthermore, k-means is known to fail clustering at very large scale, and hierarchical or approximate k-means must be used, which, in turn, tend to increase the imbalance between clusters [3]. This paper proposes an extension of the traditional k-means algorithm to produce clusters of much more even size. This is beneficial to performances since it reduces the variance and the expectation of query response times. Balancing is obtained by slightly distorting the boundaries of clusters. This, in turn, impacts the quality of results since clusters do not match the initial optimization criterion anymore. This paper is organized as follows. Section 2 defines the problem we are addressing and introduces the key metrics later used in the evaluation. Section 3 details the proposed balancing strategy. Section 4 evaluates the impact of balancing on the response times when using large collections of descriptors extracted from 1 million Flickr images. It also shows result quality remains satisfactory compared to regular k-means. Section 5 concludes the paper.

2 Problem statement

2.1 Base Clustering and Searching Methods

Without loss of generality, we partition a collection of N high-dimensional feature vectors into clusters defining Voronoi regions. We typically use a k-means algorithm quantizing the data into k clusters. This process intends to minimize the mean-squared-error (MSE) criterion, that is:

$$\sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} \|\mathbf{x} - \mathbf{c_i}\|^2, \tag{1}$$

where C_i is the i-th cluster and $\mathbf{c_i}$ is its associated centroid.

Each cluster stores a list of its associated data points. This approach is widely adopted in the context of image searches, where clustering is applied to local [13, 11] or global descriptors [2, 4]. A search strategy exploiting this partitioning is usually approximate: only one or a few clusters are probed at query time. The quality of results is typically increased when multiple clusters are probed during the search as in [7, 5, 3, 4].

The actual distances between the query point and the features stored in each such cluster are subsequently computed [1, 10]. Therefore, the response time of a query is directly related to

(i) the strategy used to identify the clusters to probe,

(ii) the total number of vectors used in distance computations.

The cost for (i) is fixed and mainly corresponds to finding the m_p centroids that are the closest to the query point (L_2). In contrast, the cost for (ii) heavily depends on the cardinality of each cluster to process. It is of course linked to m_p. Note that (i) is often negligible compared to (ii).

2.2 Metrics: Selectivity and Recall

Probing m_p clusters results in a list of vector candidates for which distance to the query vector must be computed. The trade-off between result quality and retrieval time is measured by *recall* and *selectivity*, respectively. *Selectivity* is the fraction of the data collection that is used in the distance calculations. Obviously, the larger selectivity, the higher the cost of the retrieval process. *Recall* is the average rate of nearest neighbors returned for a query, with respect to a given ground truth. In other words, recall is equal to the number of true results returned among the total set of true results. Observe that if the true nearest neighbor is found within any of the selected clusters, then it is re-ranked first if exact distance computation is performed in a subsequent stage [1, 10].

2.3 Imbalance Factor

As in [3], we empirically measure the imbalance between the cardinalities of the clusters resulting from a k-means using an *imbalance factor* γ defined as:

$$\gamma = k \sum_{i=1}^{k} {p_i}^2 \tag{2}$$

where p_i is the probability that a given vector is stored in the list associated with the i^th cluster.

As shown in [3], for a given k and $m_\mathrm{p} = 1$, γ is directly related to the search cost: $\gamma = 3$ means that the expectation of the search time is three times higher than the one associated with perfectly balanced clusters. This is the reason why imbalance factor is, to our opinion, more appropriate to assess balancing than entropy, even if both metrics reach an extremum in the case of perfectly balanced clusters.

Optimal balancing ($\gamma = 1$) is obtained when $|C_i| = N/k$ for all i. In that case, the variance of query time is zero, as any given cluster contains exactly the same number of elements, as shown by the analytical expression of the variance of the number of elements in a cluster list:

$$\mathrm{Var} = N^2 \sum_{i=1}^{k} p_i \left(p_i - \frac{1}{k}\right)^2 . \tag{3}$$

3 Balancing Clusters

3.1 The Balancing Process

Balancing clusters is an iterative post-processing step performed on the final output of a k-means type-of algorithm. The idea is to artificially enlarge the distances between the data points and the centroids of the heavily filled clusters so as to shrink and slightly drain loaded clusters. Penalties applied to distances depend on the population of clusters. Hence, the contents of clusters and thus their population can be recomputed accordingly. This balancing process eventually converges to equally filled clusters. The penalties are called *penalization terms* and are computed as follows:

$$\begin{cases} \forall i, b_i^0 = 1 \\ \forall l \leq r, b_i^{l+1} = b_i^l \left(\frac{n_i^l}{n_\mathrm{opt}}\right)^\alpha \end{cases} \tag{4}$$

where α controls the convergence speed and r is the number of iterations performed. A small value for α ensures that balancing will be done in a smooth way, while it implies to have greater r in order to get cluster of even population. Note that, at each iteration l, the populations $|C_i^l|$ are updated so as to take these penalization terms into account. More precisely, distances from any point $\mathbf{x}$ to the i^th centroid are set to

$$d_\mathrm{bal}^l(\mathbf{x}, \mathbf{c_i})^2 = d(\mathbf{x}, \mathbf{c_i})^2 + b_i^l. \tag{5}$$

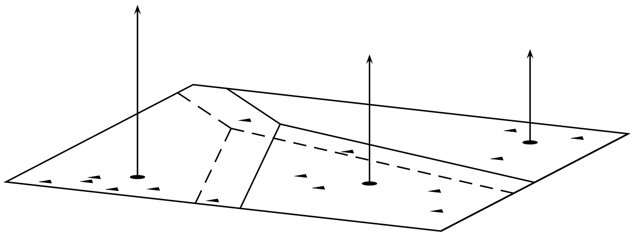

Figure 1. Data points and centroids embedded in a 3-d example. Data points are plotted as triangles while centroids are represented as dots, with a non-null z-axis value after some iterations. Initial Voronoi region boundaries and their shifted version are respectively plotted as solid and dashed lines.

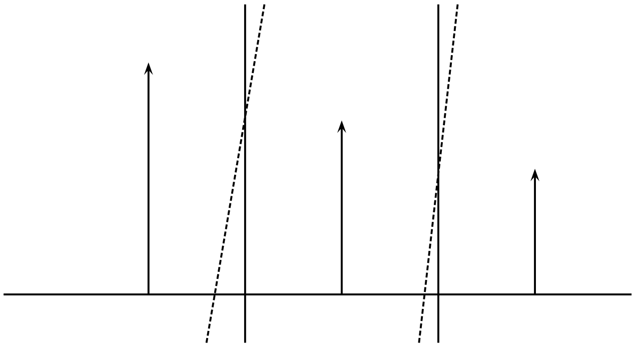

Figure 2. Voronoi region boundaries shifted after some iterations. Along iterations, clusters of large population get their centroids moved away while other centroids stay close to the $z = 0$ plane where data points lie. New boundaries, plotted as dashed lines, shrink the left-hand-side cluster because of its large population.

The initial value for b_i^0 is explained by Equation 5, as it assigns an energy to balance factors that is comparable to the one of other dimensions of the data, under the assumption that vectors are normalized.

3.2 Geometrical Interpretation

A geometrical interpretation of the balancing process described above is possible. Assume the balancing process first embeds the k-means clustered d-dimensional vectors into a $(d+1)$-dimensional space. In this space, the d first components of all vectors are the ones they had in their initial space, while component $d+1$ is set to zero. Centroids are embedded similarly, except for their last component that is set to $\sqrt{b_i^0}$. Then, along iterations, it is set to the appropriate $\sqrt{b_i^l}$ value. The intuition is that centroids are elevated in an iterative way from the hyperplane where vectors lie. The more vectors in a cluster, the more elevation its centroid gets.

This is illustrated in Figure 1, where the z-axis corresponds to the added dimension. While iterating, updated assignments for vectors are computed with respect to the coordinates of the points lying in the augmented space. The artificial elevation of centroids tends to shrink the most populated clusters, dispatching some of its points in neighboring clusters. Figure 2 exhibits the influence of the $(d+1)^{\text{th}}$ coordinate of the centroids on the position of the borders.

Note that new Voronoi region boundaries are parallel to initial ones at each step, which can be shown as follows. Updated boundary between clusters i and j is defined as

$$H_{i,j} = \left\{\mathbf{x}, f(\mathbf{x}) + b_i^l - b_j^l = 0\right\}$$

using

$$f(\mathbf{x}) = d(\mathbf{x}, \mathbf{c_i})^2 - d(\mathbf{x}, \mathbf{c_j})^2$$

where b_i^l and b_j^l are constant with respect to $\mathbf{x}$. This implies that the vector $\left(\frac{\partial f}{\partial x_1}, \cdots, \frac{\partial f}{\partial x_d}\right)$ is normal to $H_{i,j}$. As it is also normal to the initial boundary, both boundaries are parallel.

3.3 Partial Balancing

The proposed balancing strategy empirically converges towards clusters having the same cardinalities. Several stopping criteria can be applied, the most simple being a maximum number of iterations. It is also possible to target a particular value for γ, either fixed or possibly in proportion to the initial imbalance factor. Early stopping the balancing process reduces the overall distortion of the initial Voronoi regions.

4 Experiments

4.1 Datasets and Imbalance Factors Analysis

Our analysis has been performed on descriptors extracted from a large set of real-world images. We downloaded one million images from Flickr to build the database and another one thousand images for the queries. Several description schemes were applied to these images, namely

		γ	
descriptor	dimensionality	k=256	k=1024
SIFT	128	1.08	1.09
BOF	1000	1.65	1.93
GIST	960	1.72	3.75
VLAD	8192	5.41	6.23

Table 1. Imbalance factor for k-means computed on one million images descriptors of different types.

SIFT local descriptors [6], Bag-of-features [13] (BOF), GIST [12] and VLAD descriptors [4]. SIFT were extracted from Hessian-Affine regions [9] using the software of [8]. BOF vectors have been generated from these local descriptors, using a codebook obtained by regular k-means clustering with 1000 visual words. The VLAD descriptors were generated using a codebook of 64 visual words applied to the same SIFT descriptors, leading to vectors of dimension $64 \times 128 = 8192$. For GIST, we have used the most common setup (colors and 3 scales), leading to 960-dimensional descriptors. Global descriptors (BOF, GIST and VLAD) produce exactly one descriptor per image, leading to one million vectors for each type of descriptor. We have randomly sampled the SIFT local descriptors to keep 1 million vectors, so as to allow fair comparison between γ values for all types of descriptors.

In all cases, we assume a closed-world setup, i.e., the dataset to be indexed is fixed, which is valid for most applications. A more thorough study of open-world case can be found later in this section.

Table 1 reports imbalance factors obtained for each type of descriptors after performing a standard k-means clustering on our database. It can be observed that higher dimensional vectors tend to produce higher imbalance factors. BOF descriptors have an imbalance factor which is lower than GIST for a comparable dimension, which might be due to their higher sparsity. The number k of clusters has a significant impact on γ: larger values for k lead to higher γ. The low values for k we have considered here explains why γ measured for the SIFT descriptors in Table 1 are lower than those reported in the literature: [3] reports 1.21 and 1.34 for codebooks of size k=20 000 and k=200 000, respectively. Note that, in the case of hierarchical clustering, observed imbalance factor are much higher since balancing issues are present at every level of the clustering tree.

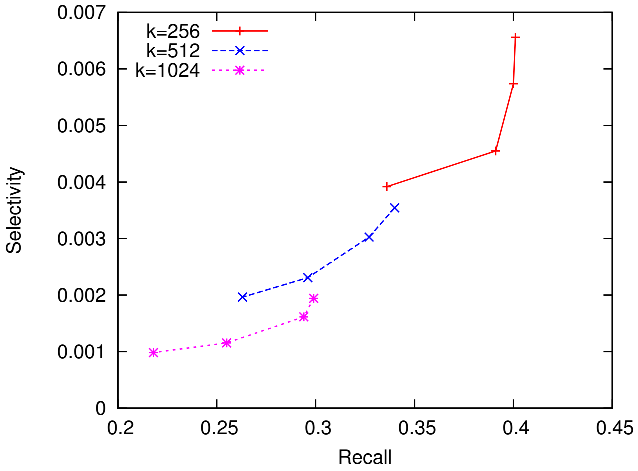

Figure 3. Impact of varying balancing on the trade-off selectivity/recall. *Top to bottom:* **the points of a given curve correspond to 0, 4, 16 and 64 iterations. Observe that first iterations of our algorithm tend to significantly lower selectivity without severely impacting recall.**

4.2 Evaluation of the proposed method

We analyze here the impact of our method on selectivity, recall and variability of the response time. So as to better study the impact of clustering in the feature space, we use a ground truth based on actual feature space distances rather than on semantic image similarity. We also analyze the convergence properties of our method. α is set to 0.01 in all our experiments. Note that our balancing strategy is especially interesting for global descriptors where having perfectly balanced clusters leads to constant query time. Therefore we performed our experimental validation on the well known BOF vectors. In the following, we use $m_\mathrm{p} = 1$ in order to better show the impact of balancing. Note however that significantly better recall performances could be achieved by using larger m_p.

4.2.1 Selectivity/recall performance

Figure 3 shows the compromise between accuracy and complexity for different values of k. First note that the trade-off between selectivity and recall can be adjusted using the number k of clusters and the number m_p of probes. Our method exhibits comparable performance with that of the k-means clustering in terms of selectivity and recall. For example, using $k = 512$, performing a k-means without balancing ($\gamma = 1.77$) leads to a selectivity of 0.0035 for a recall of 0.34, while performing only 4 iterations of our balancing strategy ($\gamma = 1.53$) allows one to achieve a selectivity of 0.0030 (-14%) for a recall of 0.33 (-3%). Note however that with our method a given selectivity/recall point is obtained with a much better (lower) variability of the response time, as shown later in this section.

4.2.2 Impact of the number of iterations

The number r of iterations in Equation 4 is an important parameter of our method, as it controls to which extent complete balancing is enforced or not. Figure 3 shows that selectivity is reduced in the first iterations with a reasonable loss in recall, i.e., comparable to what we would obtain by modifying the number of clusters. Following iterations are comparatively less profitable, as the gain in selectivity is obtained at a relatively high cost in recall. Modifying the stopping criterion allows our method to attain a target imbalance factor which is competitive with respect to the selectivity/recall trade-off.

4.2.3 Convergence speed

Figure 4 shows that even a small number of iterations can lead to reasonably balanced clusters. Full balancing is almost achieved after 64 iterations, which leads to a computational cost that is negligible in the experiments we conducted compared to that of the clustering. Higher values of k for the initial k-means do not require more balancing iterations, which is somewhat surprising as more penalization terms have to be learned. Note that convergence of our algorithm is not guaranteed, though it has been observed in our experiments.

4.2.4 Variance of the query response time

The impact of our balancing strategy on the variability of the response time is illustrated by Figure 5, which gives the distribution of the number of vectors returned by the indexing structure. The tight distribution obtained by our method shows that the objective of reducing the variability of the query time resulting from unbalanced clusters is fulfilled: the response time is almost constant with full balancing, while its variance is significantly lower with partial balancing than with the original clustering.

4.3 Is closed-world setup mandatory ?

Previous section presented results obtained in a closed-world setup as it allows to achieve quasi-constant query time in all cases. However, Figure 6 shows that, as soon as the distribution of the learning set is reasonably close to the one of the database, comparable selectivity-versus-recall compromise can be achieved in the open-world case. In this example, the database is the same as the one used in the previous experiments. For both closed-world and semi-closed-world setups, another 1 million images from Flickr are used as a learning set to train k-means. The difference between both setups is that in the semi closed-world one, balancing is learned on the database itself while in the open-world setup, it is optimized on the learning set, which could lead to unbalanced database clusters. Note nevertheless that quality of the balancing in semi closed and open-world setups strongly depends on the learning set having comparable distribution to the one of the database. Therefore, their usage should be restricted to cases where this assumption is likely to be verified, as for example in cases where the learning set is a subset of the entire database.

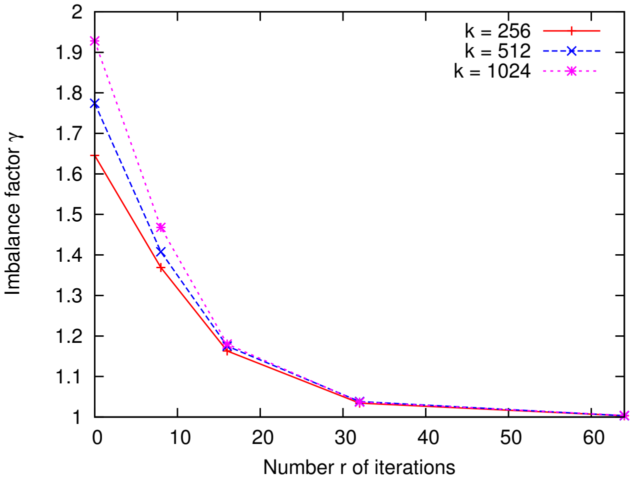

Figure 4. Convergence speed in terms of the number r of iterations.

5 Conclusion

Many high-dimensional indexing schemes partition the feature space into clusters, typically using k-means clustering. These schemes are efficient because they process a small proportion of clusters at query time. In this paper, we address an issue of these methods: having to process clusters with very different cardinalities causes important variations in the response time to queries. We propose an algorithm that iteratively balances clusters such that they become more equal in size, reducing the response time variance without significantly impacting the search quality.

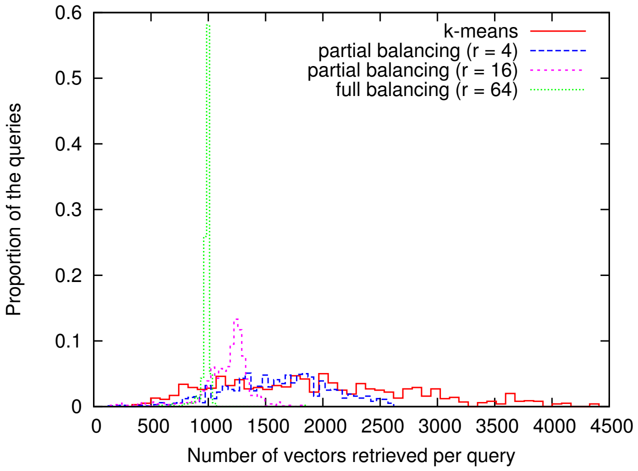

Figure 5. Histograms of the number of vectors returned, computed over our 1000 queries, for the original k-means and our algorithm with three number of iterations ($k = 1024$ in all cases). Observe the tightness of the distribution in the case of our method, which reflects a very low variability in response time.

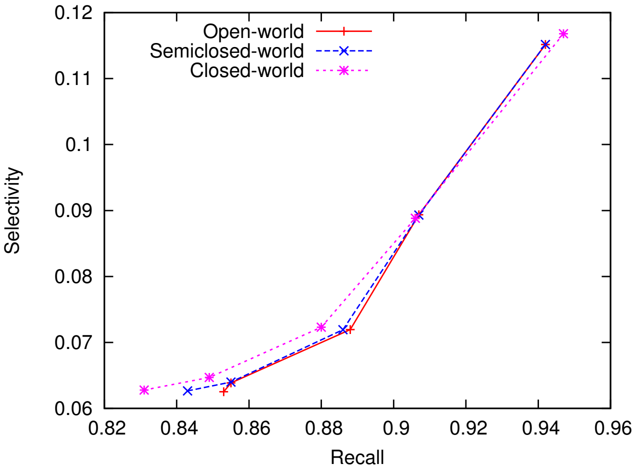

Figure 6. Compared selectivity/performance for 3 different setups. Here, we used $k = 512$ and $m_{\mathrm{p}} = 32$.

Acknowledgements

This work was partly realized as part of the Quaero Project, funded by OSEO, French State agency for innovation.

References

[1] M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the Symposium on Computational Geometry*, pages 253–262, 2004.

[2] M. Douze, H. Jégou, H. Sandhawalia, L. Amsaleg, and C. Schmid. Evaluation of gist descriptors for web-scale image search. In *Proceedings of the ACM International Conference on Image and Video Retrieval*, July 2009.

[3] H. Jégou, M. Douze, and C. Schmid. Improving bag-of-features for large scale image search. *International Journal of Computer Vision*, 87(3):316–336, February 2010.

[4] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2010.

[5] A. Joly and O. Buisson. A posteriori multi-probe locality sensitive hashing. In *Proceedings of the ACM International Conference on Multimedia*, pages 209–218, 2008.

[6] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[7] Q. Lv, W. Josephson, Z. Wang, M. Charikar, and K. Li. Multi-probe LSH: Efficient indexing for high-dimensional similarity search. In *Proceedings of the International Conference on Very Large DataBases*, pages 950–961, 2007.

[8] K. Mikolajczyk. Binaries for affine covariant region descriptors, 2007.

[9] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.

[10] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *Proceedings of the International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, 2009.

[11] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2161–2168, 2006.

[12] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001.

[13] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1470–1477, 2003.