Computer Vision

Approximate Nearest Neighbors Algorithms

Vladislav Belov

FNSPE CTU

March 28, 2020

ANN - Approximate Nearest Neighbors

In this brief report, we investigate fast nearest neighbors algorithms (in particular, the library called FLANN). FLANN includes such algorithms as *k-d forest* and *priority search k-means tree* [3].

Notes on Implementation

Experiments were performed in Python 3.8.1, using the PyFLANN library. Technical details are available on our GitHub repository.

What did we do?

1. In the scope of this work, naive implementation of the k-means [2] clustering algorithm was performed by means of assigning points to clusters using various techniques for nearest neighbors search (see the respective GitHub page):
   - k-d forest;
   - priority search k-means tree;
   - exact assignment.
2. Clusters were always initialized the same way: randomized initialization by sampling from the uniform distribution $U(\alpha, \beta)$ with constant seed where α and β are given by values in the data set (minimum and maximum, respectively).[1]No normalization was involved.
3. Afterwards, we tested the functionality of our implementation on synthetic data (see Fig. 1) and also its ability to label points for a SIFT data set with approx. 2M records.

[1] This is only one of many solutions - more optimal ones are available, e.g. k-means++ [1].

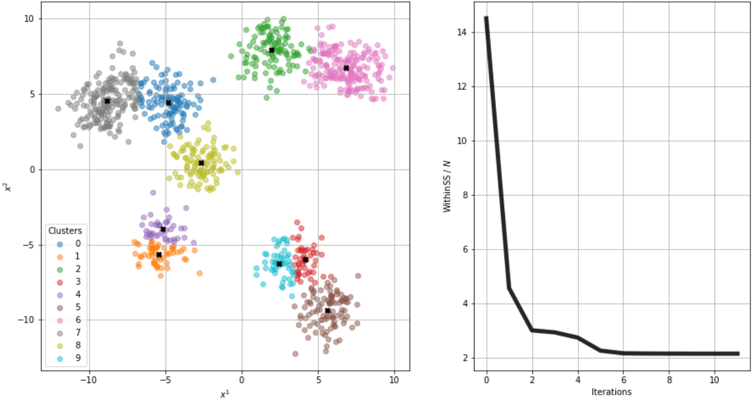


Figure 1: Naive k-means implementation output with points labeled using the priority search k-means tree algorithm.

Framework of Experiments

- For each NN technique applied on the SIFT data set, we calculate time spent on computations per iteration and visually compare behavior of the cost function $C = \mathrm{WithinSS}/N = \frac{1}{N}\sum_{i=1}^{N}\|X_i - f(X_i)\|_2^2$. The number of cluster centers was set to 32000; each run continued for 30 iterations.
- **Notation**:
  - $X \in \mathbb{R}^{N\times d}$ where N is the number of samples and d is the dimension (in our case, $N = 2097152$, $d = 128$);
  - X_i, $i \in \{1, 2, \ldots, N\}$, is the i-th row of matrix X;
  - $f(X_i)$ denotes the coordinates of the cluster center associated with X_i;
  - $\|\cdot\|_2$ is the Euclidean norm $\left(\|\cdot\|_2 = \sqrt{\langle\cdot,\cdot\rangle}\right)$.

Runs

Due to bounded computational capabilities, we tested two sets of parameters for approximate nearest neighbors techniques and measured time to evaluate NN and time to assign new cluster centers within individual iterations:

- k-d forest (*T* - number of randomized trees, *C* - number of leaves to check in one search):

ID	T	C	(AVG. | STD.) NN time, [s]	(AVG. | STD.) assignment time, [s]
1	32	75	197.11 | 16.8	225.1 | 15
2	64	100	323.39 | 32.35	212.65 | 21.76

- Priority search *k*-means tree (*B* - branching factor for *k*-means tree construction, *I* - number of iterations for the tree construction, *CB* - cluster boundary index for the search in the tree, *C* - number of leaves to check in one search):

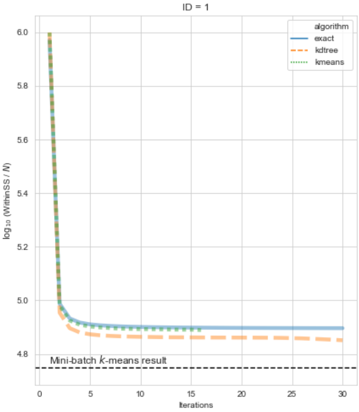
ID	B	I	CB	C	(AVG. | STD.) NN time, [s]	(AVG. | STD.) assignment time, [s]
1	32	20	0.2	75	102.2 | 39	220.6 | 7.29
2	64	20	0.2	100	137.07 | 22.14	236.47 | 13.22

Exact Point Assignment

- With $N \approx 2\text{M}$, $d = 128$, and the number of cluster centers equal to 32000, the problem is infeasible for short time frames.
- We tried to assign points directly by evaluation of distances to all cluster centers and soon halted the calculation, as $\frac{1}{4}$ of the first iteration took approx. 4 hours (30 iterations would result in approx. 20 days).
- Another tested approach was assignment of points to respective centers by means of standard k-d trees:

(AVG. | STD.) NN time, [s]	(AVG. | STD.) assignment time, [s]
9085.98 | 4160.11	202.04 | 14.03

- For the sake of being able to provide at least some margin for comparison of results, we have also performed mini-batch *k*-means clustering [4] with same initial cluster centers. It took the algorithm from 6 to 8 hours to converge in 890 iterations. The resulting value of the cost function is equal to 55864.63 (for more information, visit here).



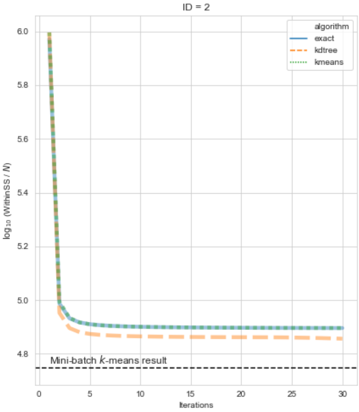


Figure 2: Comparison of different approximate nearest neighbors evaluation approaches in the scope of k-means iterations.[2]

[2] On the left-hand-side, the priority search k-means tree approach stops at iteration 16 due to the fact that the cost function stopped decreasing and no progress was made during the following 10 iterations.

The experiment on the previous slide leads to the following conclusions:

- For the given set of parameters and data set, the priority search k-means tree is faster than the k-d forest for the cost of being less accurate. Nonetheless, its accuracy is comparable to that of a k-d forest approach.
- Convergence of ANN-based methods is highly dependent on initial parameters and poor parameter choice might prevent them from progress towards the optimal solution. Examples from Fig. 2 show that convergence of the k-means method might be faster with ANN search than that with exact assignment.

Even though the convergence of k-means might be faster with ANN assignment during first iterations due to shuffling of poorly labeled points, the cost function is no longer guaranteed to be optimized (see Fig. 3). The guarantee vanishes by cause of the fact that labels are no longer optimal before cluster centers rearrangement. To prevent the cost function from increasing, we introduce simple post-processing of ANN-based labels:

- For points whose new labels are different from that of the previous iteration, we check if their newly assigned centers are indeed closer than the previous ones. If not, old labels are retained.

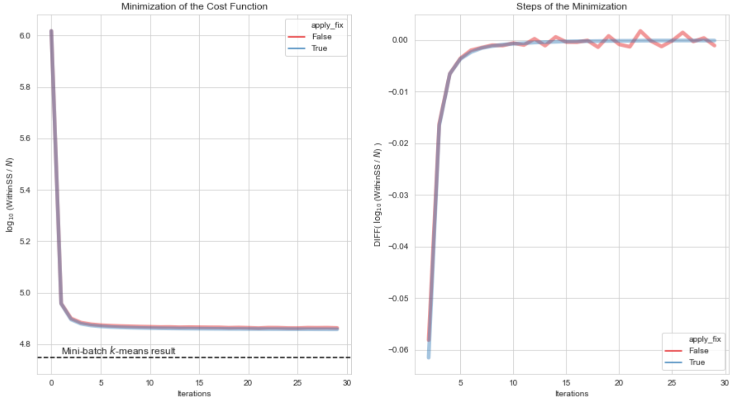


Figure 3: Performance of the priority search k-means tree with and without the proposed post-processing of labels. The diagram confirms that the cost function can increase when the ANN-based label assignment is used. Moreover, it shows that the proposed post-processing procedure fixes the problem. Used parameters:

Fix	B	I	CB	C	(AVG. | STD.) NN time, [s]
False	8	10	0.2	25	63.06 | 11.1
True	8	10	0.2	25	63.51 | 8.61

Appendix

D. Arthur and S. Vassilvitskii.
K-means++: The advantages of careful seeding.
In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, page 1027–1035, USA, 2007. Society for Industrial and Applied Mathematics.

S. Lloyd.
Least squares quantization in pcm.
IEEE Transactions on Information Theory, 28(2):129–137, March 1982.

M. Muja and D. Lowe.
Fast approximate nearest neighbors with automatic algorithm configuration.
VISAPP 2009 - Proceedings of the 4th International Conference on Computer Vision Theory and Applications, 1:331–340, Jan 2009.

D. Sculley.
Web-scale k-means clustering.
In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, page 1177–1178, New York, NY, USA, 2010. Association for Computing Machinery.