Computer Vision

Approximate Nearest Neighbors Algorithms
Usage of GPU

Vladislav Belov

FNSPE CTU

June 6, 2020

Contents

Fair AI Similarity Search (`faiss`)

In this report, we cover results of our experiments with `faiss`[1] library [2] for the nearest neighbors search. The report includes the following:

- Timing comparison for the k-means task on SIFT2M data evaluated before;
- k-NN graph construction with various indices from `faiss` and evaluation of timings and precision on Oxford105K data set.
- Implementation of the smooth path algorithm within the k-NN graph based on (12) of [2]. Oxford105K data was utilized.

Technical Details

The respective implementation is available on our GitHub page.

[1] Link to the Fair AI Similarity Search library we used: faiss.

Contents

k-means Exercise Revisited with GPU

Once again we perform naive k-means iterations on the SIFT2M data set with k $= 32000$. However, this time during each iteration only exact nearest neighbors are searched for. Moreover, the search is done by means of GPU. As expected, the `IndexFlatL2` index from `faiss` on a GPU-enabled machine performs the search of nearest cluster centers more than a thousand times faster (recall that it took approx. 9K seconds before). Cluster assignment still takes relatively long, as it was not optimized for GPU by us.

Title	(AVG. | STD.) NN time, [s]	(AVG. | STD.) cluster assignment time, [s]
k-means iteration	2.58 | 0.37	136.56 | 18.36

Contents

k-NN Graph Construction with GPU

We also used the power of GPU and `faiss` to construct k-NN graphs. A few indices were tried out: `IndexFlatL2` (exhaustive searches), `IndexIVFFlat` (non-exhaustive searches with inverted indices), and `IndexIVFPQ` (product quantization powered with a coarse quantizer). The table below contains fixed parameters used to measure both timings to construct k-NN graphs and precision of results:

- Fig. 1 demonstrates timings dependent on the query size (note that the current GPU-enabled implementation of `faiss` (version 1.6.3) only scales up to 1024 neighbors per query);
- Fig. 2 showcases performance of `IndexIVFFlat` and `IndexIVFPQ`. To estimate performance, we measure recall which is calculated as the average fraction over queries (all images from the data set) of first k true nearest neighbors found during the search of K approximate nearest neighbors, $k \leq K$; during our experiments, we put $K = 100$.

Index	nlists	M	nbits	nprobe
`IndexFlatL2`	-	-	-	-
`IndexIVFFlat`	256	-	-	32
`IndexIVFPQ`	256	16	8	32

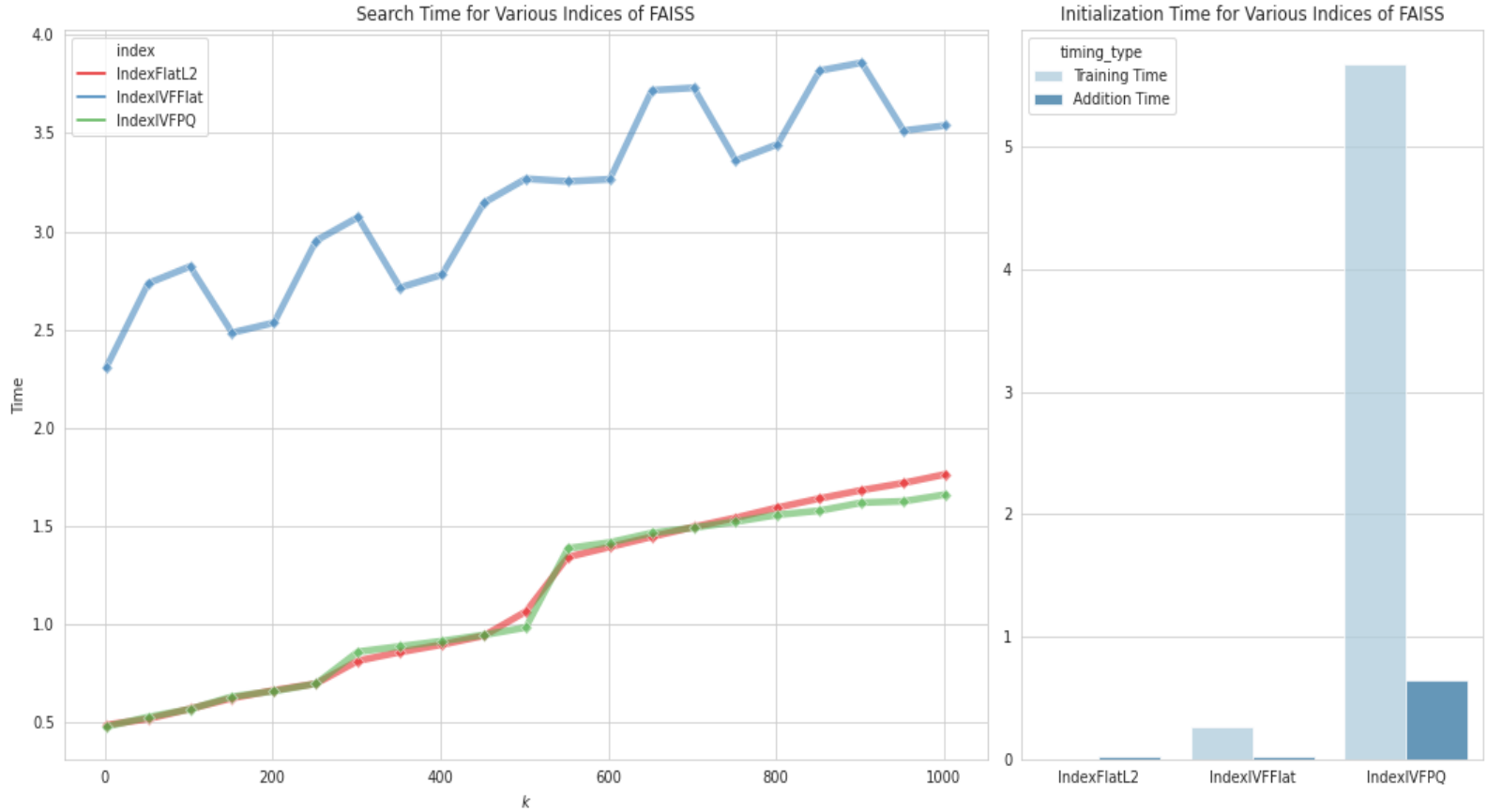

Figure 1: Timings for k-NN graph construction (query time on the left-hand side, index training and point addition time on the right-hand side) on Oxford105K data set. It is clear that, in terms of time, `faiss` powered by GPU outperforms KGraph [1] we experimented with before (recall that it took KGraph approx. 105s to finish the search for 100 nearest neighbors).

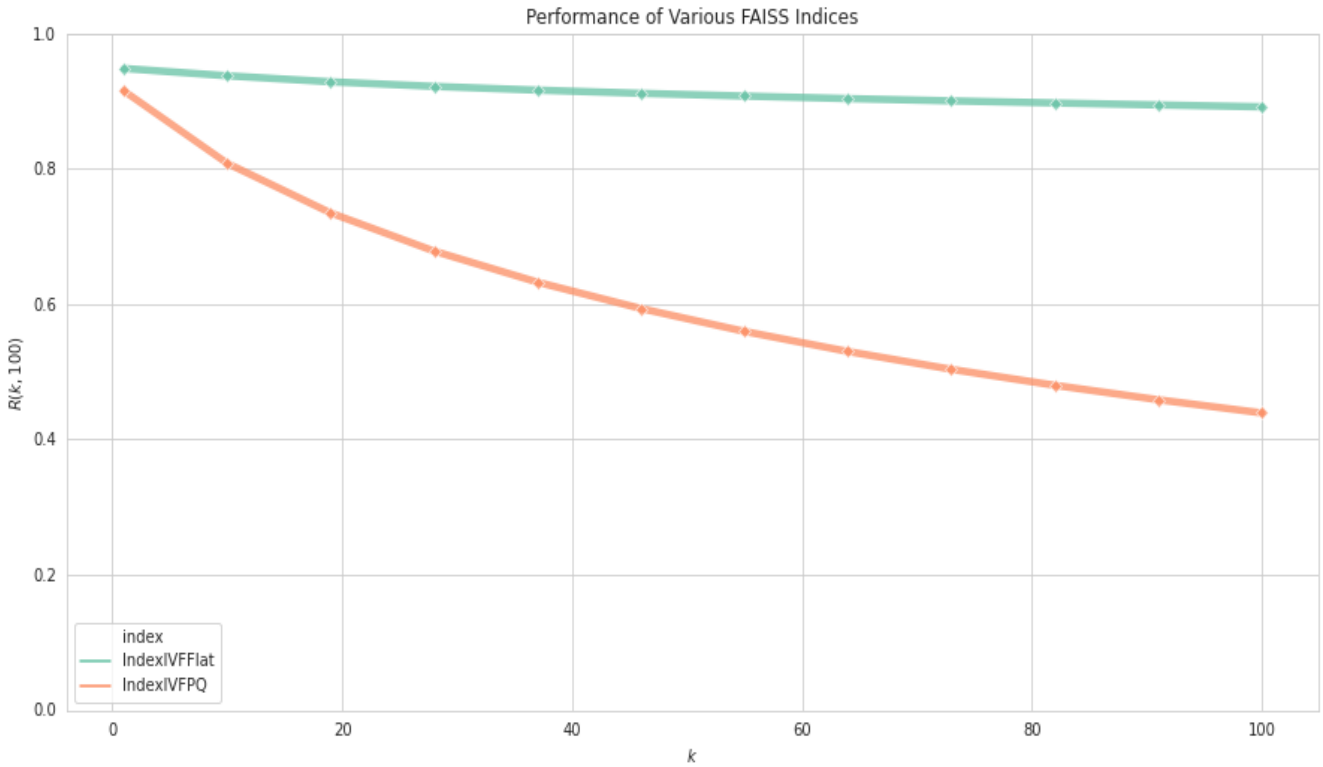

Figure 2: R(k, 100) measured for `IndexIVFFlat` and `IndexIVFPQ` on Oxford105K data set.

Smooth Path Evaluation

Another exercise we have performed in the scope of this work is the implementation of the path search given by equation (12) in [2]. We have utilized a depth-first search with a cut-off to generate simple paths in the k-NN graph. To stress the preference on path smoothness, the choice of the final path was made according to cost function $\min_P \max_i d_{p_i,p_{i+1}}$ where $P = (p_1, p_2, \ldots, p_n)$ denotes paths between the source and the target and $n <$ cut-off value. If more than one paths can be chosen, the shortest one of the remaining is picked. An example is shown in Fig. 3.

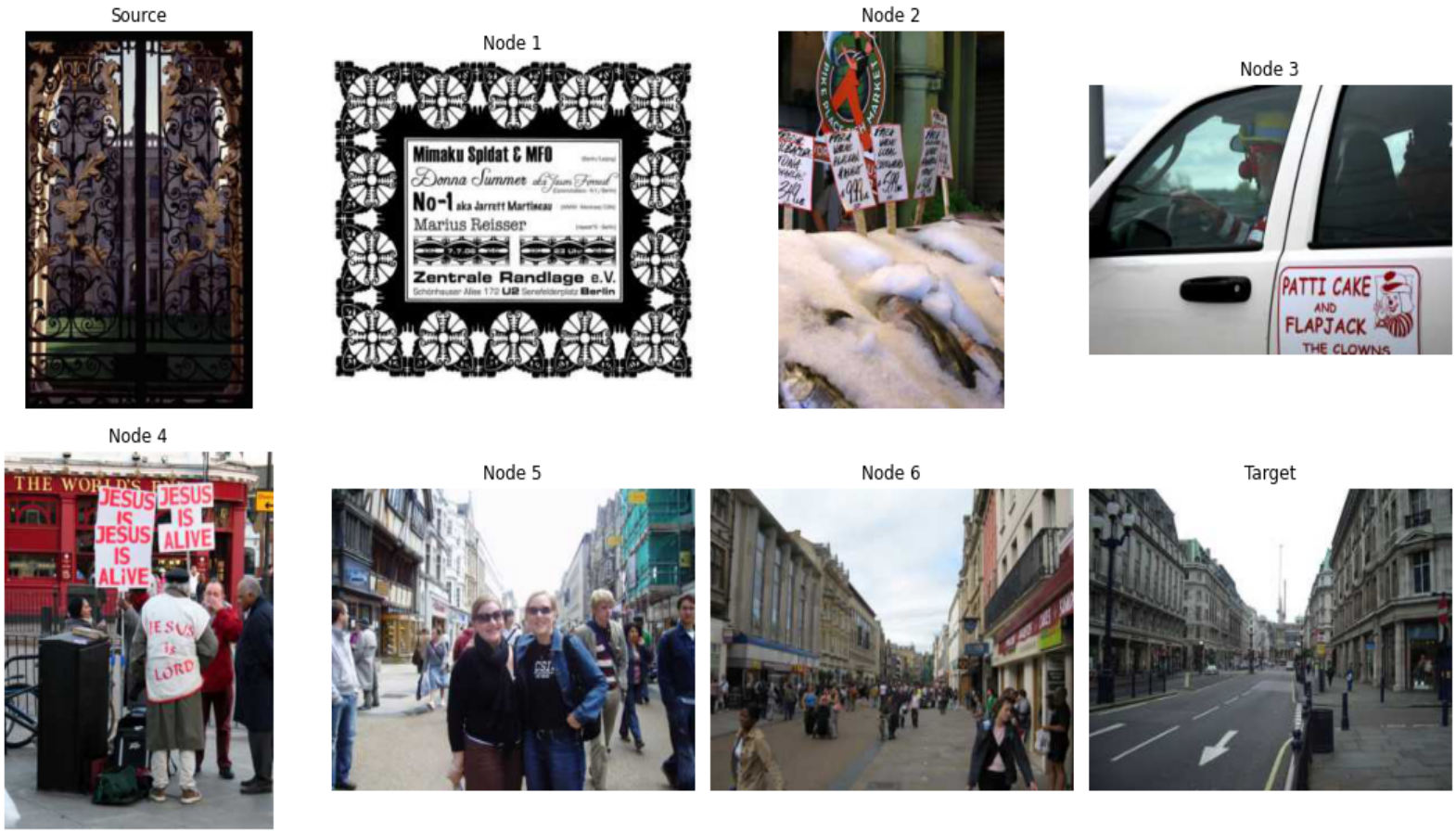

Figure 3: Sample of the smooth path search between the source image and the target image within the exact 10-NN graph of the whole Oxford105K data set; the cut-off value was set to 7.

Appendix

W. Dong, C. Moses, and K. Li.

Efficient k-nearest neighbor graph construction for generic similarity measures.
In Proceedings of the 20th International Conference on World Wide Web, WWW '11, page 577–586, New York, NY, USA, 2011. Association for Computing Machinery.

J. Johnson, M. Douze, and H. Jégou.

Billion-scale similarity search with gpus.
IEEE Transactions on Big Data, pages 1–1, 2019.