Efficient Large-scale Approximate Nearest Neighbor Search on the GPU – Supplementary Material –

Patrick Wieschollek[1,4] Oliver Wang[2] Alexander Sorkine-Hornung[3]
Hendrik P.A. Lensch[1]

[1] University of Tübingen [2] Adobe Systems Inc. [3] Disney Research
[4] Max Planck Institute for Intelligent Systems, Tübingen

Re-Ranking The line projection approach for re-ranking proposed nearest neighbor candidates uses a lossy compression of the original datapoints, which projects each vector $x \in \mathbb{R}^D$ to a vector $\tilde{x} \in \mathbb{R}^{P_{\text{line}}}$. When choosing the correct value of P_{line} usually the tradeoff between better approximation (higher values of P_{line}) and less memory consumption (smaller values of P_{line}) arises. To illustrated the effect of our re-ranking approach with $P_{\text{line}} \ll D$, we applied PQT with re-ranking to the MNIST dataset of handwritten-digests $\mathcal{X} \subseteq \mathbb{R}^{784}$ using $P_{\text{line}} = 28$ (see Figure 2), where $\mathcal{L}_c$ comes from the bin traversal and $\mathcal{L}_s$ is the result after re-ranking.

Distribution of Data While the SIFT1M dataset contains vectors in the hypercube $x \in [0, 255]^{128}$, these vectors are not embedded in the entire space, i.e., the intrinsic dimension is much lower. Figure 1 visualizes a random subset of SIFT1M using *t-Stochastic-Neighbor-Embedding (t-SNE)*, which reveals two clusters. A uniform $[0, 255]^{128}$ distributed dataset would be visualized as a uniform 2D point cloud by t-SNE.

Distribution of vectors Figure 4 visualizes a projection of the Voronoi-cells from the PQT trained on the SIFT1M dataset. Mapping a database vector $x \in \mathcal{X}$ to one of $N = (C_1 \cdot C_2)^P$ bins allows us to prune the entire search space by only picking bins during a query which are likely to contain the nearest neighbor. However, those bins may contain millions of vectors when N is small. On the other hand, traversing million of bins is infeasible, even on the GPU. Figure 3 contains the bin-histogram for the SIFT1B dataset (Table 1) in combination with hashing for a maximum of 100M bins. Using higher values of N allows a much finer granularity of produced bins.

Notice, by restricting the number of bins to be at most 100M by hashing, bins are unions of different clusters. A re-ranking step would only pick vectors from the correct cluster, because of the smaller approximate distance.

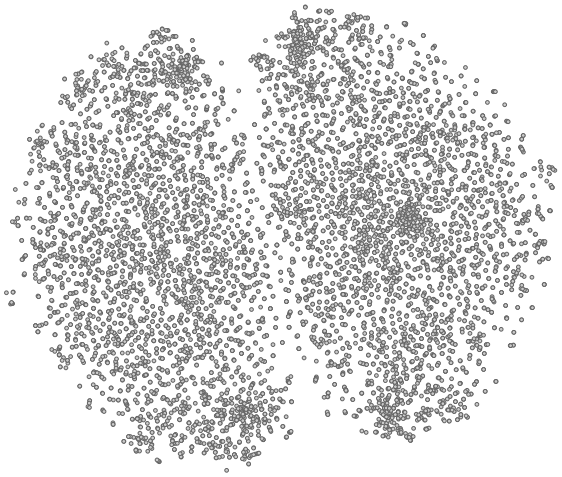

Figure 1: Visualization of 100K randomly chosen points from SIFT1M using t-SNE reveals two clusters of the data, such that the data is not fully embedded in the space.

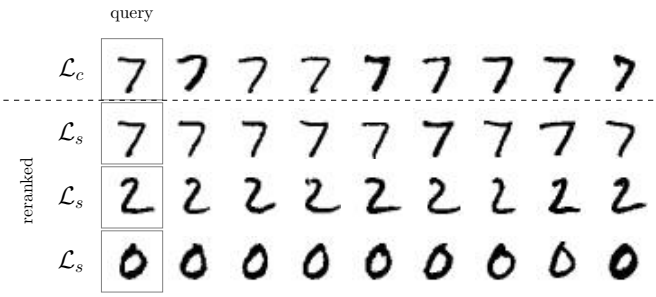

Figure 2: Each row visualizes the list of nearest neighbor candidates $\mathcal{L}_s$ (resp. $\mathcal{L}_c$) for the training MNIST dataset for a query from the test set below (resp. above) the dashed line. Re-ranking these $D = 784$ dimensional raw image vectors was done with our line projection method ($P_{\text{line}} = 28$).

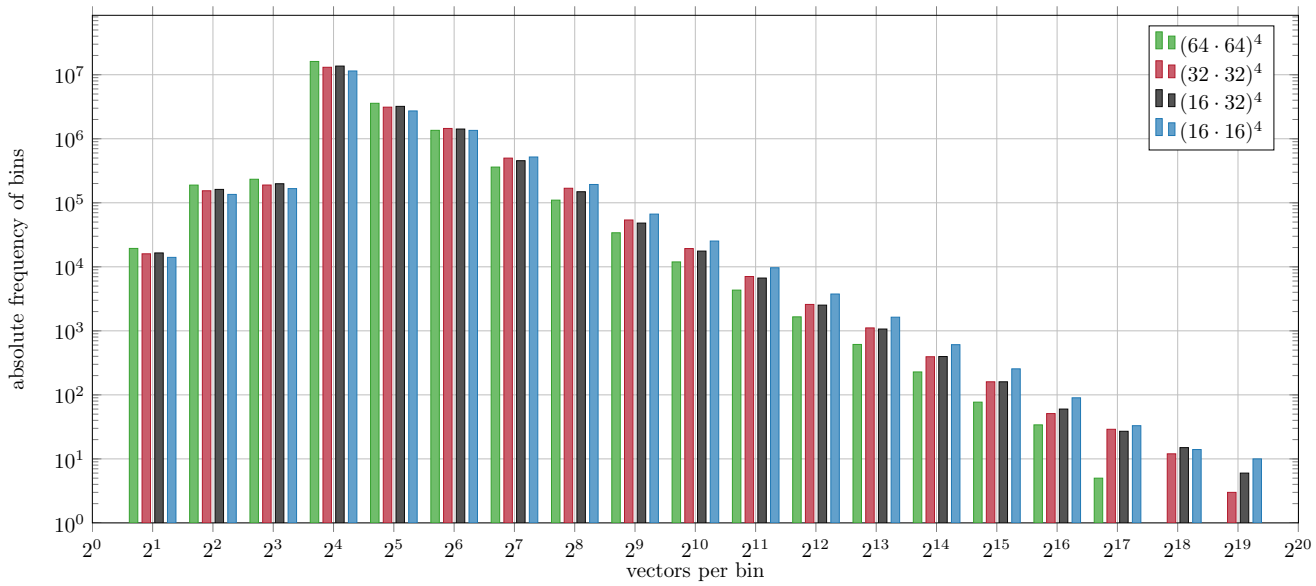

Figure 3: The SIFT1B dataset is split up into $\max\{10^8, (C_1 \cdot C_2)^4\}$ bins by a PQT with hashing. Each bar above 2^k counts the number of bins which contains $[2^{k-1}, 2^k]$ vectors. Note, how more cluster centers yield fewer highly occupied bins. Distributions with smaller median and shorter tail are better.

	absolute frequency of bins			
vectors per bin	$(64 \cdot 64)^4$	$(32 \cdot 32)^4$	$(32 \cdot 16)^4$	$(16 \cdot 16)^4$
2	19,381	15,963	16,473	14,044
4	189,062	154,028	161,860	135,162
8	233,141	189,513	198,391	166,573
16	16,122,413	13,076,618	13,654,802	11,451,256
32	3,581,330	3,115,776	3,206,650	2,720,408
64	1,354,592	1,449,036	1,418,639	1,351,564
128	361,308	498,769	454,400	519,172
256	110,007	168,674	148,859	193,192
512	34,012	53,860	48,198	66,601
1,024	11,925	19,287	17,610	25,269
2,048	4,327	7,056	6,678	9,670
4,096	1,654	2,585	2,521	3,761
8,192	613	1,111	1,067	1,634
16,384	228	394	397	608
32,768	77	160	160	255
65,536	34	51	60	90
131,072	5	29	27	33
262,144	0	12	15	14
524,288	0	3	6	10
1,048,576	0	0	0	8
2,097,152	0	0	0	0

Table 1: Numerical values of Figure 3

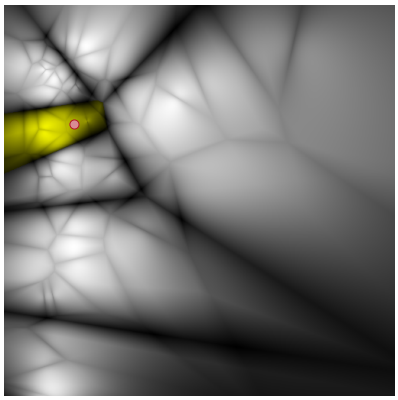

Figure 4: A PQT tree trained on SIFT1M and projected to 2D using random coordinates. Each pixel represents a point p. The gray-value is defined as $0.8f_1 + 0.2f_2$, where f_k is the ratio between the smallest two distances from p to a centroid from the k-th layer in the PQT tree to emphasize the hierarchy. To find nearest neighbor candidates for a query ● the PQT prunes the search space to the highlighted area ($w = 1$), which itself is then further clustered.