Algorithm Engineering for High-Dimensional Similarity Search Problems

Martin Aumüller
IT University of Copenhagen, Denmark
maau@itu.dk

Abstract

Similarity search problems in high-dimensional data arise in many areas of computer science such as data bases, image analysis, machine learning, and natural language processing. One of the most prominent problems is finding the k nearest neighbors of a data point $q \in \mathbb{R}^d$ in a large set of data points $S \subset \mathbb{R}^d$, under same distance measure such as Euclidean distance. In contrast to lower dimensional settings, we do not know of worst-case efficient data structures for such search problems in high-dimensional data, i.e., data structures that are faster than a linear scan through the data set. However, there is a rich body of (often heuristic) approaches that solve nearest neighbor search problems much faster than such a scan on many real-world data sets. As a necessity, the term *solve* means that these approaches give approximate results that are close to the true k-nearest neighbors. In this talk, we survey recent approaches to nearest neighbor search and related problems.

The talk consists of three parts: (1) What makes nearest neighbor search difficult? (2) How do current state-of-the-art algorithms work? (3) What are recent advances regarding similarity search on GPUs, in distributed settings, or in external memory?

2012 ACM Subject Classification Theory of computation → Nearest neighbor algorithms; Computing methodologies → Simulation evaluation; Theory of computation → Computational geometry

Keywords and phrases Nearest neighbor search, Benchmarking

Digital Object Identifier 10.4230/LIPIcs.SEA.2020.1

Category Invited Talk

1 Difficulty of nearest neighbor search (NNS)

In the first part of the talk, we give an overview over the general setting of high-dimensional NNS. We look at general observations about the difficulty of high-dimensional NNS, a phenomenon usually referred to as the "concentration of distances" [5]. This for example happens if we draw data and query points at random from a d-dimensional standard normal distribution $\mathcal{N}(0,1)^d$; for large d, the distance to the furthest neighbor will be very close to the distance of the nearest neighbor. Next, we look at the benchmarking tool `http://ann-benchmarks.com` [2]. We observe that there exist many different approaches that solve NNS with almost perfect quality – in terms of the fraction of true nearest neighbors found on average – with a query time that is several orders of magnitude faster than a linear scan through the dataset. Before looking at these approaches in detail, we survey approaches to measure the difficulty of solving a NNS task discussing the work of [3, 10].

2 Current state-of-the-art approaches to NNS

In the second part of the talk, we explore the landscape of NNS implementations. The focus of the discussion will be on hashing-based approaches using locality-sensitive hashing and graph-based approaches.

Approaches using locality-sensitive hashing [12] are excellent candidates for designing theoretical sound data structures for NNS. In a nutshell, a locality-sensitive hash function makes it possible to map a data point to a number, such that the smaller the distance between the points, the more likely it is that they are mapped to the same number. The talk puts an emphasis on solving k-NNS using a recent adaptive query algorithm on an LSH data structure proposed in [4], building on the general idea of adaptive query algorithms discussed in [1]. We review the details of the query algorithm and engineering choices on the way, such as hash functions pools and sketches as described in [6], and fast distance estimation using AVX instructions.

In contrast to the theoretical guarantees that can be achieved using LSH approaches, graph-based approaches give little theoretical guarantees (a notable exception is the analysis in [15]). Nonetheless, implementations following this approach dominate performance benchmarks. The idea of a graph-based approach is to consider each data point in the data set as a node in a graph. An edge (u, v) represents that v is among the nearest neighbors of u. Given a query point, a greedy search – usually starting with a random start node – is performed to obtain a candidate set of nearest neighbors. Graph-based approaches have several complications that need to be addressed. For example, in high-dimensional space we usually find a very skewed degree distribution which leads to large hubs that make the search slow [19]. Another problem lies in the random starting node: finding a good neighborhood of nodes requires the insertion of long-range edges and diversification of node neighborhoods. The talk surveys recent approaches addressing these issues [13, 16].

As a potential starting point for future engineering work, we further discuss the recent machine learning approach [7] that considers NNS as a learning problem and proposes to use a learned index in the style of [14].

3 Similarity search problems on the GPU, in external memory, and in distributed settings

In the last part of the talk, we survey recent progress on similarity search problems in other areas. We discuss how graph-based algorithms are made to run efficiently on GPUs [9] and in external memory [20]. Finally, we discuss recent work in the context of *similarity joins*, in which we are given two datasets $R, S \subset \mathbb{R}^d$ and want to emit all pairs $(u, v) \in R \times S$ such that the similarity between u and v is above a given threshold value. This problem is of particular interest, because a recent survey [8] found that distributed implementations using MapReduce on a small cluster are usually worse than non-distributed implementations working on a single core [17]. We will discuss general difficulties of similarity joins and take a look at a recent near-optimal solutions based on LSH [11, 18].

References

1 Thomas D. Ahle, Martin Aumüller, and Rasmus Pagh. Parameter-free locality sensitive hashing for spherical range reporting. In *SODA*, pages 239–256. SIAM, 2017.

2 Martin Aumüller, Erik Bernhardsson, and Alexander John Faithfull. ANN-Benchmarks: A benchmarking tool for approximate nearest neighbor algorithms. *Inf. Syst.*, 87, 2020. See https://arxiv.org/abs/1807.05614 for an open access version.

3 Martin Aumüller and Matteo Ceccarello. The role of local intrinsic dimensionality in benchmarking nearest neighbor search. In *SISAP*, volume 11807 of *Lecture Notes in Computer Science*, pages 113–127. Springer, 2019.

4 Martin Aumüller, Tobias Christiani, Rasmus Pagh, and Michael Vesterli. PUFFINN: parameterless and universally fast finding of nearest neighbors. In *ESA*, volume 144 of

LIPIcs, pages 10:1–10:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. URL: `https://github.com/puffinn/puffinn`.

5 Kevin S. Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. When is "nearest neighbor" meaningful? In *ICDT*, volume 1540 of *Lecture Notes in Computer Science*, pages 217–235. Springer, 1999.

6 Tobias Christiani. Fast locality-sensitive hashing frameworks for approximate near neighbor search. In *SISAP*, volume 11807 of *Lecture Notes in Computer Science*, pages 3–17. Springer, 2019.

7 Yihe Dong, Piotr Indyk, Ilya Razenshteyn, and Tal Wagner. Learning space partitions for nearest neighbor search. In *International Conference on Learning Representations*, 2020.

8 Fabian Fier, Nikolaus Augsten, Panagiotis Bouros, Ulf Leser, and Johann-Christoph Freytag. Set similarity joins on MapReduce: An experimental survey. *PVLDB*, 11(10):1110–1122, 2018.

9 Fabian Groh, Lukas Ruppert, Patrick Wieschollek, and Hendrik Lensch. Ggnn: Graph-based gpu nearest neighbor search. *arXiv preprint arXiv:1912.01059*, 2019.

10 Junfeng He, Sanjiv Kumar, and Shih-Fu Chang. On the difficulty of nearest neighbor search. In *ICML*. icml.cc / Omnipress, 2012.

11 Xiao Hu, Ke Yi, and Yufei Tao. Output-optimal massively parallel algorithms for similarity joins. *ACM Transactions on Database Systems*, 44(2):1–36, April 2019. `doi:10.1145/3311967`.

12 Piotr Indyk and Rajeev Motwani. Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, pages 604–613, New York, NY, USA, 1998. ACM. `doi:10.1145/276698.276876`.

13 M. Iwasaki and D. Miyazaki. Optimization of Indexing Based on k-Nearest Neighbor Graph for Proximity Search in High-dimensional Data. *ArXiv e-prints*, October 2018. `arXiv:1810.07355`.

14 Tim Kraska, Alex Beutel, Ed H Chi, Jeffrey Dean, and Neoklis Polyzotis. The case for learned index structures. In *Proceedings of the 2018 International Conference on Management of Data*, pages 489–504, 2018.

15 Thijs Laarhoven. Graph-based time-space trade-offs for approximate near neighbors. In *Symposium on Computational Geometry*, volume 99 of *LIPIcs*, pages 57:1–57:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

16 Yury A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 42(4):824–836, 2020.

17 Willi Mann, Nikolaus Augsten, and Panagiotis Bouros. An empirical evaluation of set similarity join techniques. *Proceedings of the VLDB Endowment*, 9(9):636–647, May 2016. `doi:10.14778/2947618.2947620`.

18 Samuel McCauley and Francesco Silvestri. Adaptive MapReduce similarity joins. In *BeyondMR@SIGMOD*, pages 4:1–4:4. ACM, 2018.

19 Milos Radovanovic. Hubs in nearest-neighbor graphs: Origins, applications and challenges. In *WIMS*, pages 5:1–5:4. ACM, 2018.

20 Suhas Jayaram Subramanya, Devvrit, Rohan Kadekodi, Ravishankar Krishnaswamy, and Harsha Simhadri. DiskANN: Fast accurate billion-point nearest neighbor search on a single node. In *NeurIPS 2019*, November 2019. URL: `https://www.microsoft.com/en-us/research/publication/diskann-fast-accurate-billion-point-nearest-neighbor-search-on-a-single-node/`.