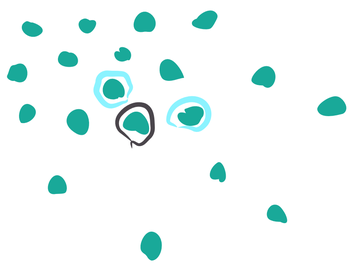
A Theory-Based Evaluation of Nearest Neighbor Models Put Into Practice

Hendrik Fichtenberger and Dennis Rohde

Part of Data Processing Pipeline

given: point-set in euclidean space



build

nearest neighbor model (black box)

input

query: give k nearest points to

output

not directly measurable

our solution

What is the quality of the output?

Query Complexity

measure of efficiency

fixed here

d-dimensional kissing number

number of points

Testing k-nearest Neighborhood

- $O(k^2 \cdot \nu_\varepsilon \cdot n \cdot \psi_d^2)$ queries sufficient in bounded average-degree graphs
- $\Omega(n/(\varepsilon k) + k \cdot \psi_d)$ queries required in general graphs

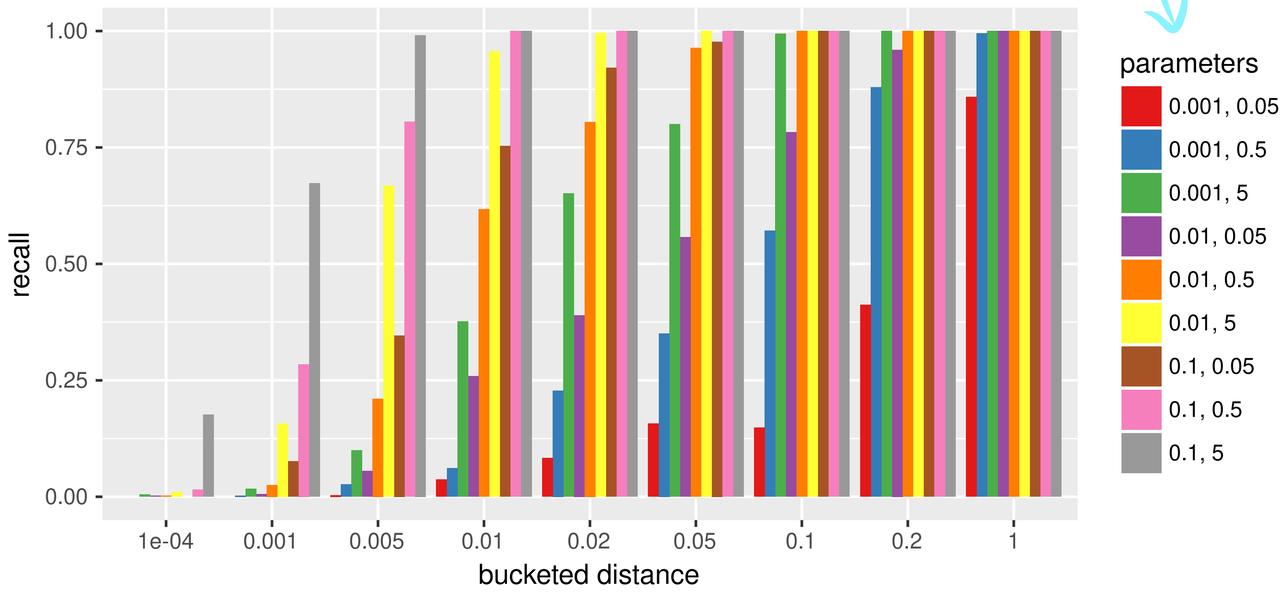
Experiments

all models were built ten times for each parameterization, then tested once

recall of algorithm

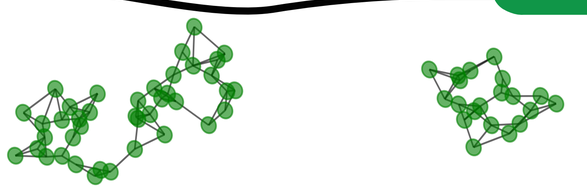
by ε-distance of model

constants of O-notation

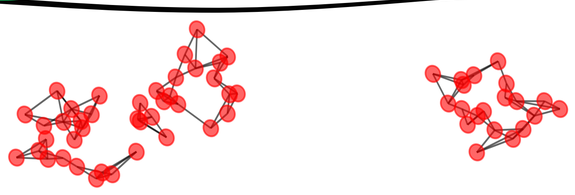


datasets: MNIST, Fashion-MNIST, Sift models: KGraph(Ann Arbor Algorithms), hnsw and SWGraph(NMSLib)

Example Graphs



brute force 3-nn graph



graph from Annoy(Spotify) model

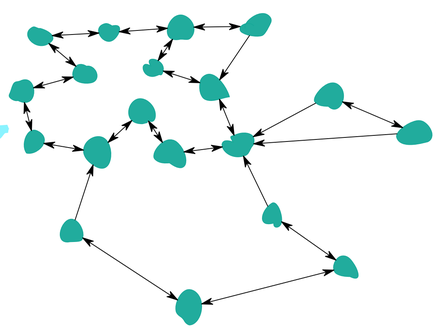
60 vertices are incident with faulty edges

Our Algorithm

is a

has query-access to

1. implicit conversion of k-nn model to geometric graph
   - no need to compute full graph



vertices = points
edges = query results

2. test if given graph is k-nearest neighbor graph
   - directed edges
   - regular (out-degree = k)
   - edges point to k-nearest neighbors of vertex

Property Testing Algorithm

- accepts every k-nearest neighborhood graph with high probability
- rejects every graph that is ε-far from being a k-nearest neighborhood graph with high probability
  - at least an ε-fraction of edges are faulty
- can freely decide otherwise

Our Algorithm

- sample $O(k \cdot 1/\varepsilon \cdot \sqrt{n})$ vertices uniformly at random
  - throw away vertices with high degree
- sample $O(k \cdot \psi_d \cdot \sqrt{n})$ vertices uniformly at random
- for every vertex in first sample check if any vertex from second sample lies nearer than any neighbor

Yes → Reject

Never → Accept

more details on arxiv

140 random points
420 edges
4 clusters

Our Code:

The algorithm: github.com/derohde/knn_test
Extension of ann-benchmarks:
github.com/hfichtenberger/ann-benchmarks

Models:

AAALgo: github.com/aaalgo/kgraph
NMSLIB: github.com/nmslib/nmslib
Annoy: github.com/spotify/annoy

technische universität dortmund
European Research Council
Established by the European Commission