
Elastiknn
Elasticsearch Plugin for Nearest Neighbor Search

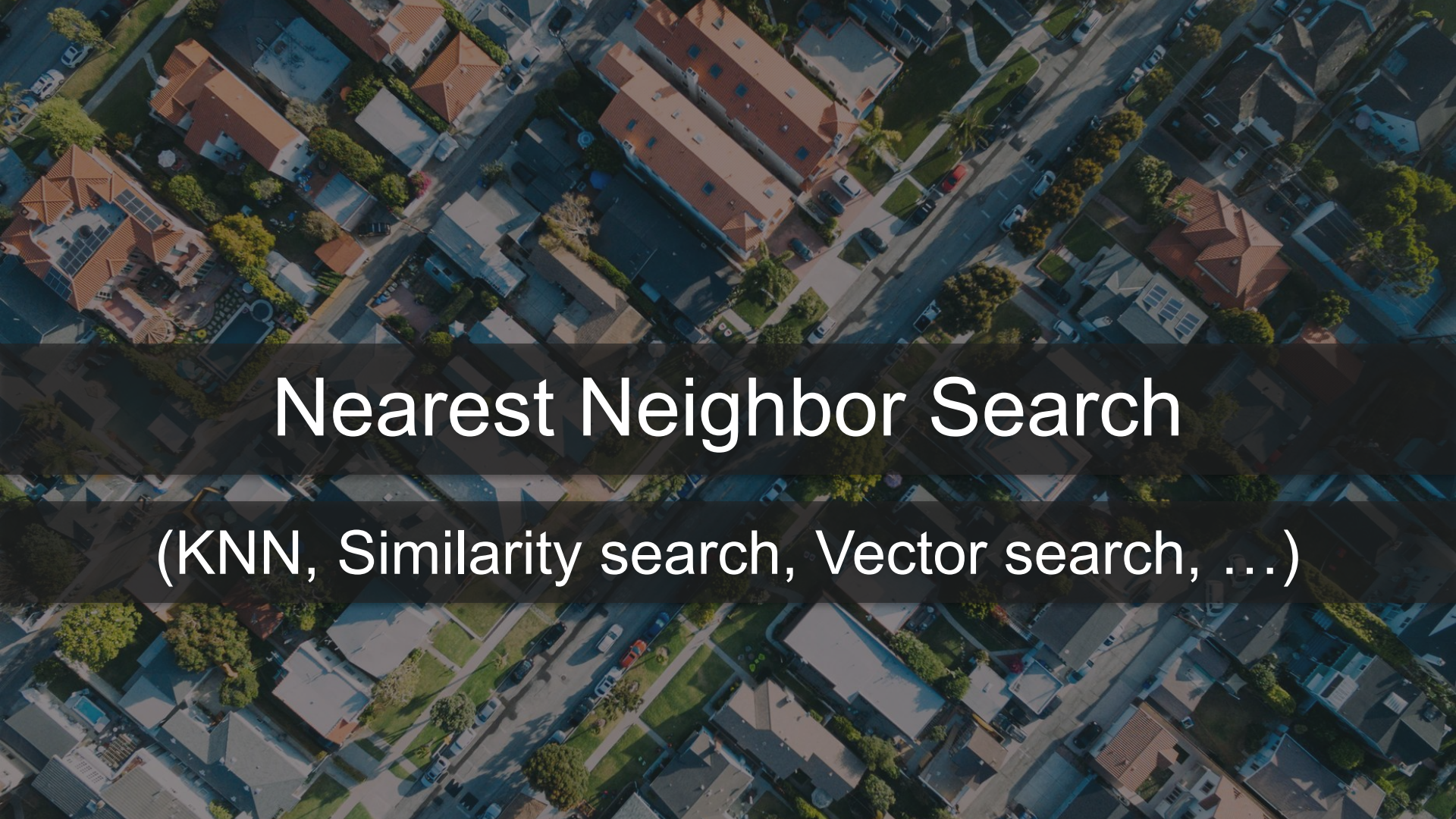
Alex Klibisz

- Software and infrastructure engineer at CiBO Technologies
  - Carbon credit marketplace for farmland
  - Primarily work on web services and data pipelines using Scala, Postgres, and Kubernetes
- Nearest neighbor search hobbyist

Overview

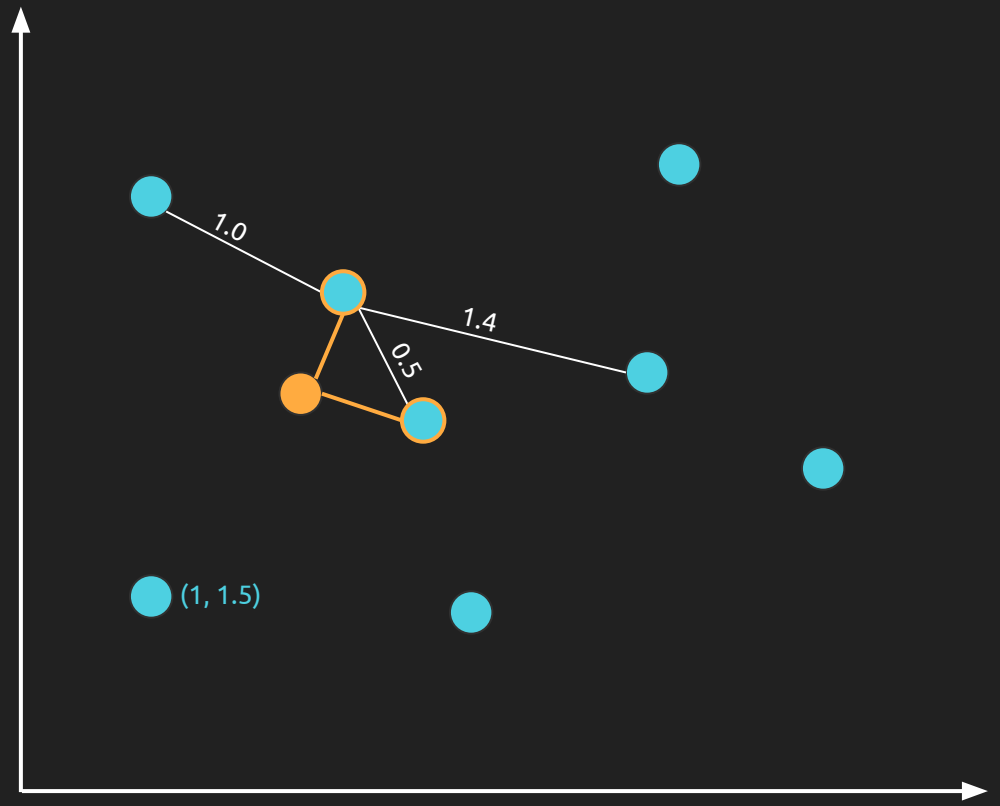
1. Nearest neighbor search
2. Elastiknn Live Demo
3. Performance, Trade-offs, Alternatives

~30 minutes followed by Q&A



Nearest Neighbor Search

(KNN, Similarity search, Vector search, …)


1.0
1.4
0.5
(1, 1.5)

- Generalizes to vectors with many dimensions (~10 to 10000)
- Vectors represent discrete entities (users, items, images, videos, etc.)
- Distance corresponds to semantic distance
  - Two items w/ similar properties should have smaller distance than two very different items
  - A user should be closer to an item s/he prefers than to an item they don't
  - Several ways to compute distance: Euclidean, Angular, Manhattan, Jaccard, Hamming
  - Similarity is the inverse of distance
- Exact KNN is usually too slow
  - Scales with the size of dataset
  - Approximate methods tradeoff search quality and speed

Elastiknn Demo

Amazon Reviews Dataset

- Properties, image vectors, reviews for ~9M Amazon products
  - Image vectors computed via convolutional neural network
- Combine standard Elasticsearch queries with nearest neighbors queries

Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering; R. He, J. McAuley; WWW, 2016
Image-based recommendations on styles and substitutes; J. McAuley, C. Targett, J. Shi, A. van den Hengel; SIGIR, 2015

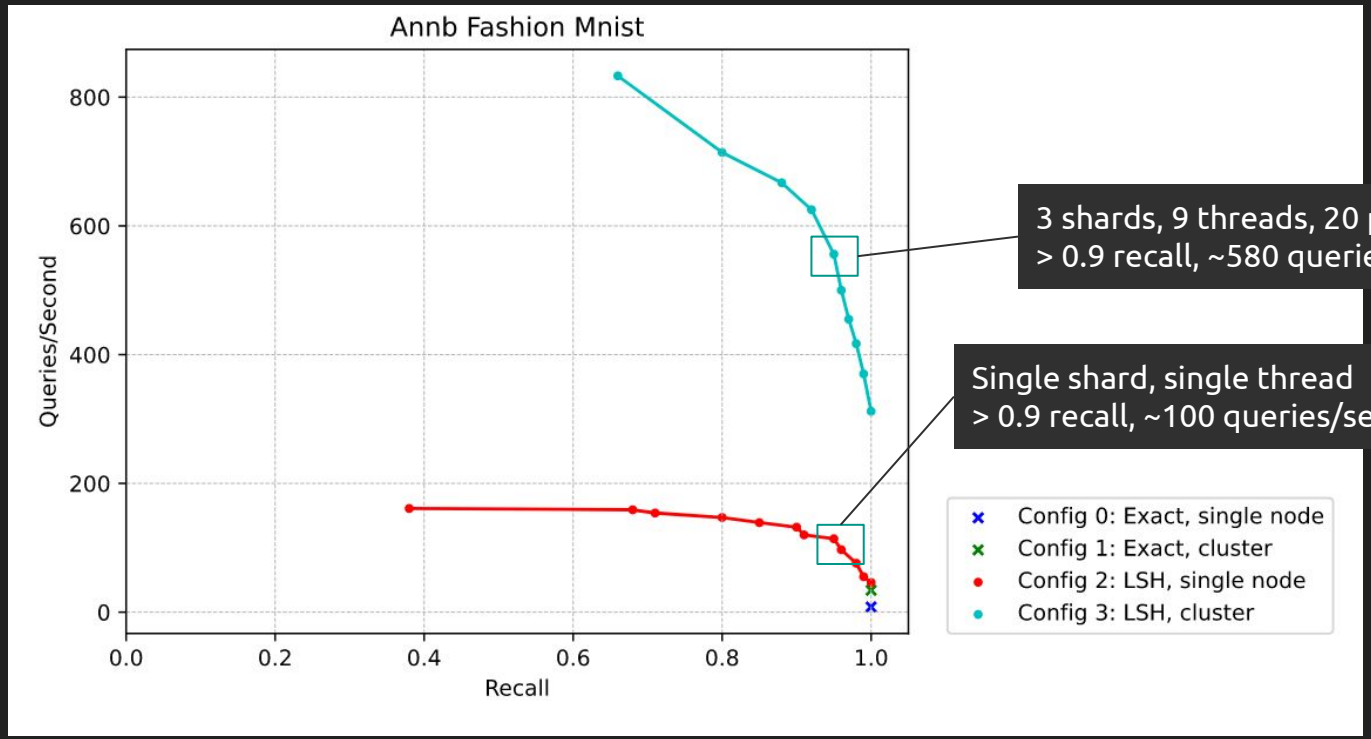
Demo...

Elastiknn Functionality Summarized

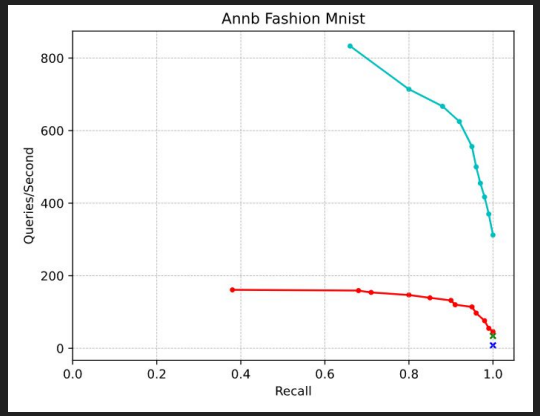
1. Store dense floating point and sparse boolean vectors
2. Exact and approximate nearest neighbors
    a. Dense vectors: L2 (Euclidean), L1 (Manhattan), Angular
    b. Sparse vectors: Jaccard, Hamming
3. Integrate with standard ES queries
4. Changes to vectors reflected immediately in search results.
5. Runs entirely in the Elasticsearch JVM

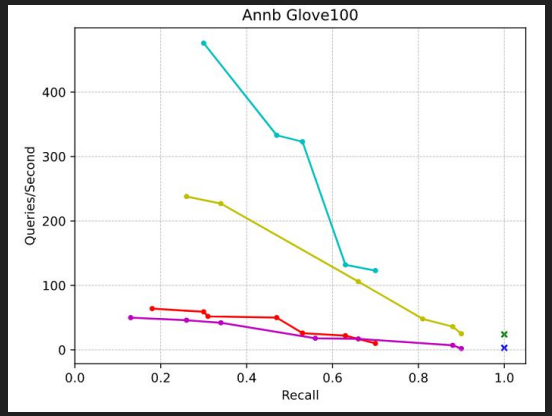
Performance, Alternatives, Trade-offs

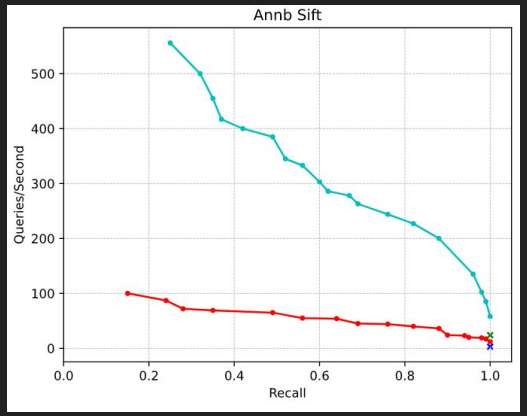
Performance = recall vs. queries/second for a specific dataset



elastiknn.com/performance







Alternatives

- Elasticsearch X-Pack
  - Supports exact queries on dense vectors since 7.3.0
  - No approximate queries
  - Tolerable latency *if* you can narrow down to ~10k vectors using other properties

- Opendistro K-NN Plugin
  - Supports exact and approximate queries for Euclidean and Angular similarity
  - ~3x faster than Elastiknn
  - Stores vectors and runs queries using NMSLib sidecar binary

Related Work

- Stackoverflow: similar image search by pHash distance in Elasticsearch (Sep 2015)
- Paper: Semantic Vector Encoding and Similarity Search Using Fulltext Search Engines (Jun 2017)
- Plugin: Elasticsearch vector scoring (Sep 2017, deprecated)
- Stackoverflow: query nearest neighbors of a point in Elasticsearch (Oct 2017)
- Paper: Towards Practical Visual Search Engine within Elasticsearch (June 2018)
- Paper: Large-Scale Image Retrieval with Elasticsearch (June 2018)
- Plugin (POC): ElastiK Nearest Neighbors (July 2018, deprecated)
- Elasticsearch PR: LSH for approximate nearest neighbour search (July 2019, closed)
- Elasticsearch 7.3: using vectors in document scoring (Aug 2019)
- Paper: Lucene for Approximate Nearest-Neighbors Search on Arbitrary Dense Vectors (Oct 2019)
- Plugin: Elastiknn (Feb 2020)
- Plugin: Opendistro ANN (April 2020)
- Lucene: Various additions to natively support vectors and graph-based search (ongoing)

Elastiknn Literature Review (Google doc)

Trade-offs

Pros

1. You don’t need a separate nearest neighbor search system
2. Changes are reflected immediately in search results

Cons

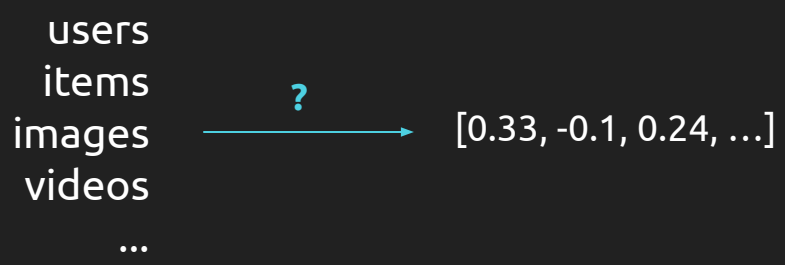
1. Order of magnitude slower than offline/batch methods
    a. Elastiknn still handles 100s of queries/second
    b. See ann-benchmarks for fastest overall solutions

Summary

Summary

- Elastiknn brings exact and approximate KNN to Elasticsearch
- Store and search dense and sparse vectors w/ five similarity functions
- Apache 2.0 License, accepting issues and PRs
- Future topics
  - How does it actually work?
  - A few interesting JVM/Lucene performance optimizations

github.com/alexklibisz/elastiknn, www.elastiknn.com



- Vector space modeling on music data (iHeartRadio)
- StarSpace (Facebook)
- Word2Vec Tutorial (Chris McCormick)
- Stop Using Word2Vec (StictthFix, Chris Moody)
- Listing Embeddings in Search Ranking (Airbnb)
- Approximate Nearest Neighbors and Vector Models (Erik Bernhardsson)
- Pinterest Visual Search

	Elasticsearch (x-pack)	Opendistro K-NN	Elastiknn
JVM-only	✓		✓
Dense L2, Exact	✓	✓	✓
Dense L2, Approx		✓	✓
Dense Angular, Exact	✓	✓*	✓
Dense Angular, Approx		✓*	✓
Dense L1, Exact	✓		✓
Dense L1, Approx			
Sparse Jaccard, Exact	✓**		✓
Sparse Jaccard, Approx			✓
Sparse Hamming, Exact	✓**		✓
Sparse Hamming, Approx			✓

* experimental ** deprecated