# VLDB 2020 Tutorial

# Similarity Query Processing for High-Dimensional Data


Jianbin Qin
Shenzhen Institute of Computing Sciences
Shenzhen University

Wei Wang
University of New South Wales

Chuan Xiao
Osaka University and Nagoya University

Ying Zhang
University of Technology Sydney

# Outline



- Introduction
- Exact Query Processing
- Approximate Query Processing
- Selectivity Estimation
- Open Problems

# Introduction

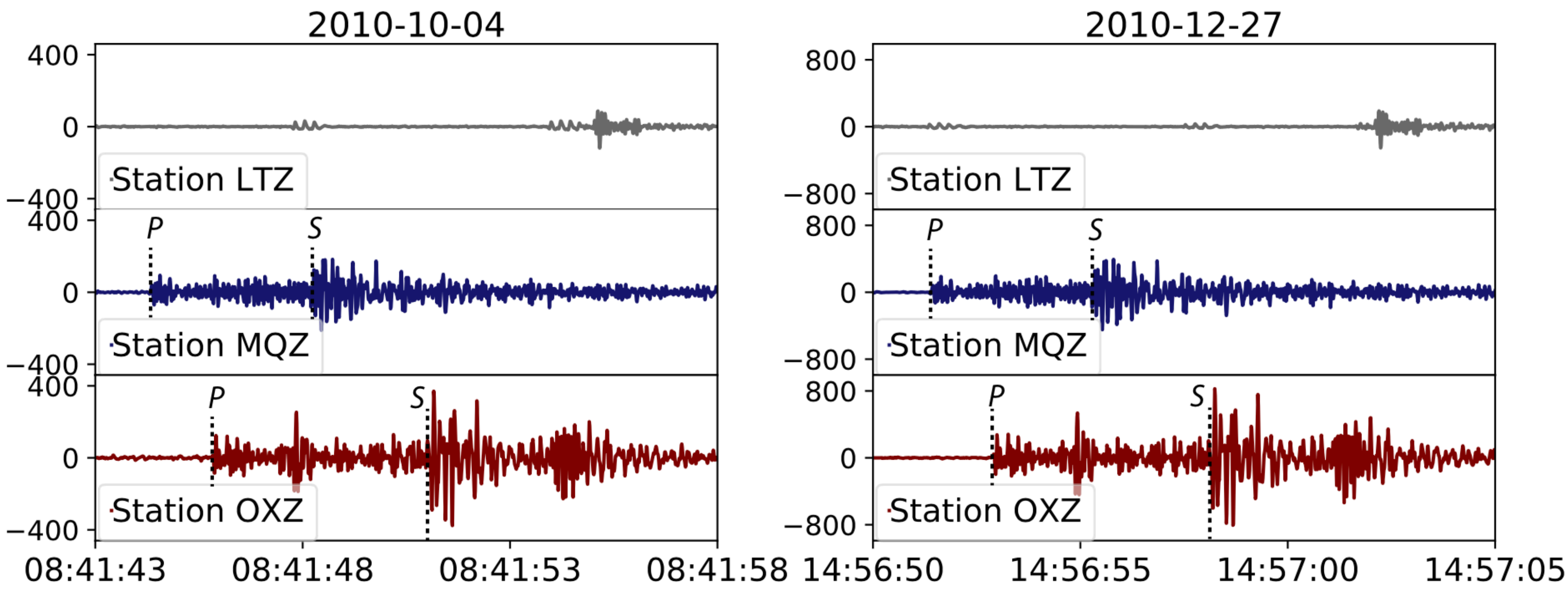


- High-dimensional data is abundant
  - Traditional sources:
    - **Time-series** [EZPB19], scientific applications
    - Document, multimedia, strings, feature vectors
  - New data sources:
    - Embedding from deep learning models
- Growing size and complexity
  - Web, social network, IoT
  - NOAA (USA) collects 100TB sensing data / day for weather forecasting
  - A variety of similarity/distance functions concerned

# Example: Scientific Applications



- High-dimensional data in huge volumes in scientific domains [YHEB17, RYBE+18]

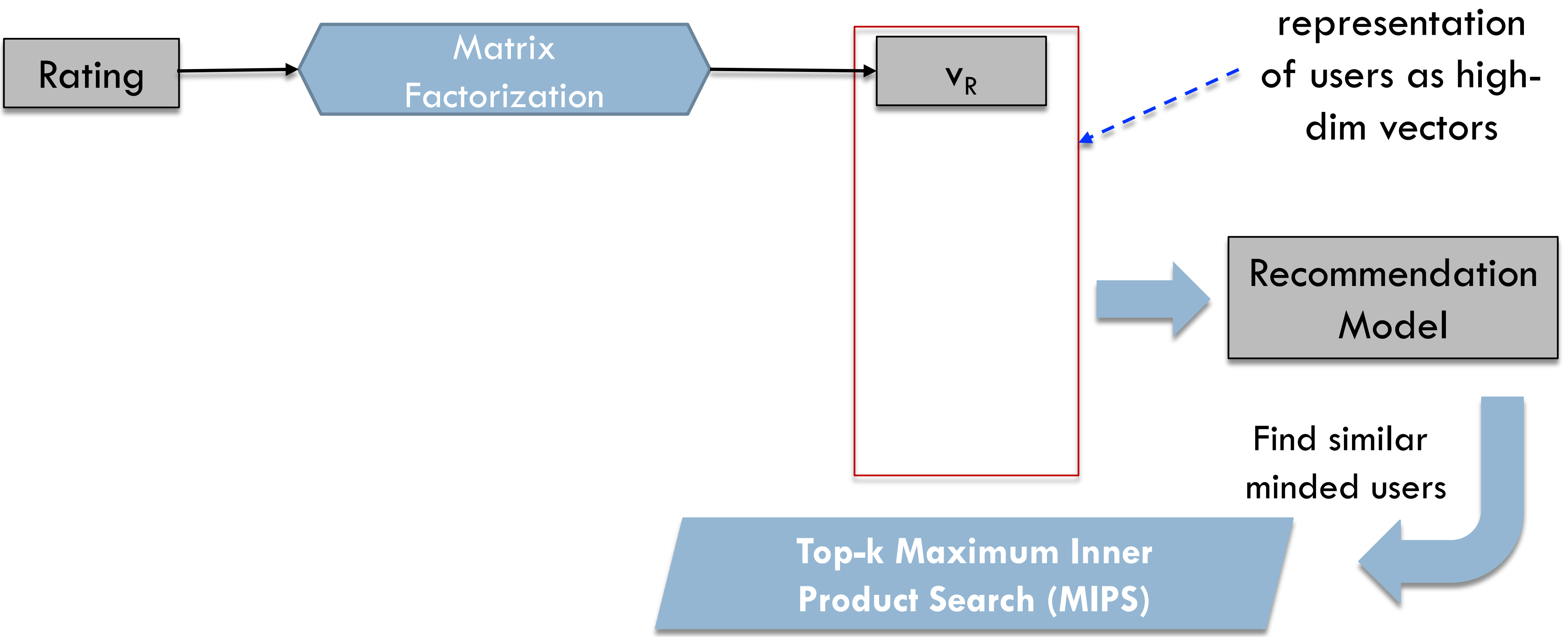



**Research Question**: Whether the magnitude 4.7 earthquake in Arkansa 2011 was caused by wastewater injection

# Example: Embedding Vectors



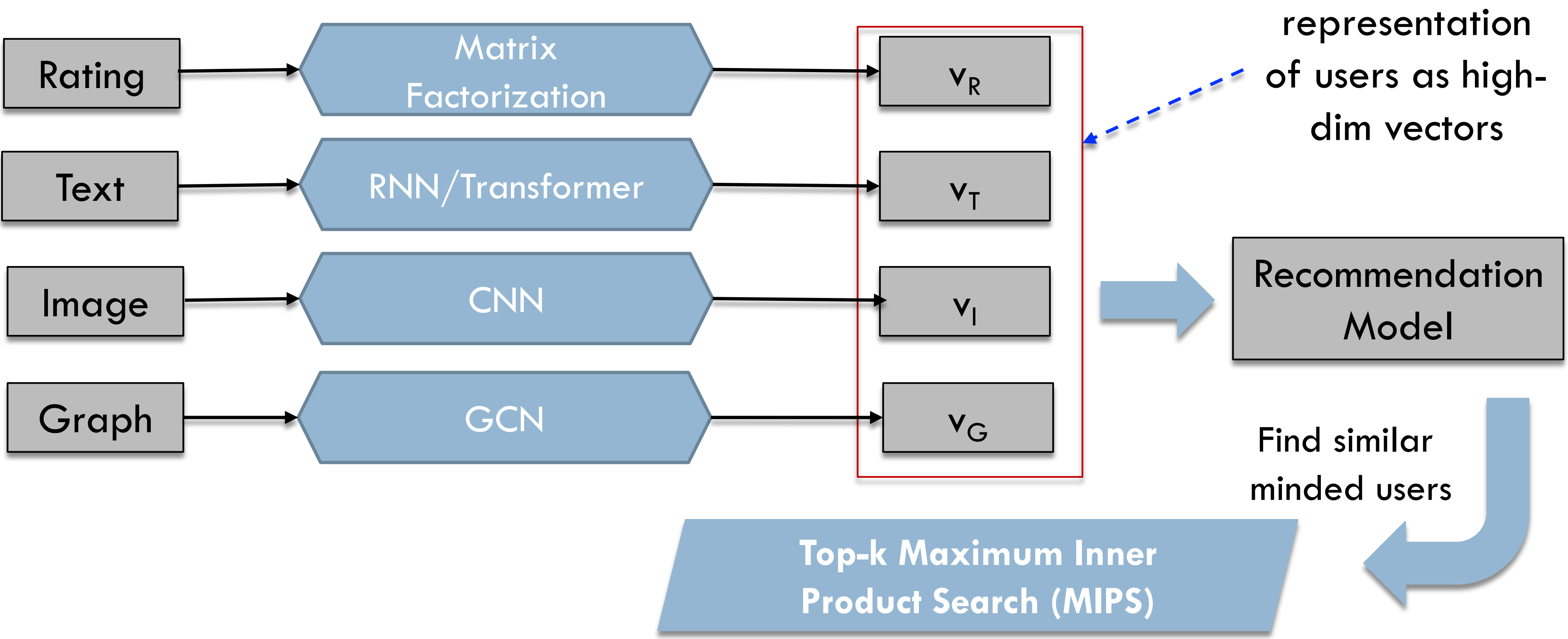
- DL presents a unified and engineering-friendly way to handle various information sources
  - Representation learning: e.g., embedding



# Example: Embedding Vectors



- DL presents a unified and engineering-friendly way to handle various information sources
  - Representation learning: e.g., embedding



# Example: Usage in Machine/Deep Learning



- Kernel trick
  - $\varphi$: mapping low-dim feature vectors to high-dim vectors

$$\langle \varphi(x), \varphi(x') \rangle = \mathcal{K}(x, x')$$

- Feature hashing trick
  - $\varphi$: **random** mapping high-dim feature vectors to low-dim vectors

$$\mathbb{E}\left[\langle \varphi(x), \varphi(x') \rangle\right] = \langle x, x' \rangle$$

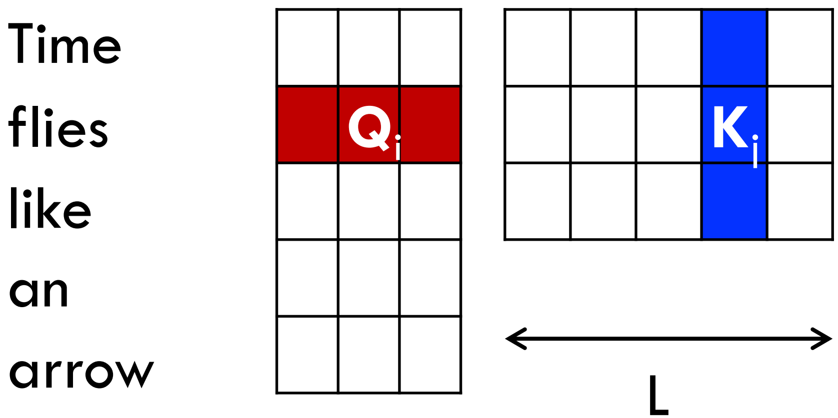
Improves efficiency, scalability and sometimes effectiveness

# Example: Usage in Machine/Deep Learning



- Reformer [KKL20]
  - Speed up self-attention

$$\mathrm{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}$$




Find batch top-k $\mathbf{K}_i$'s for each $\mathbf{Q}_i$

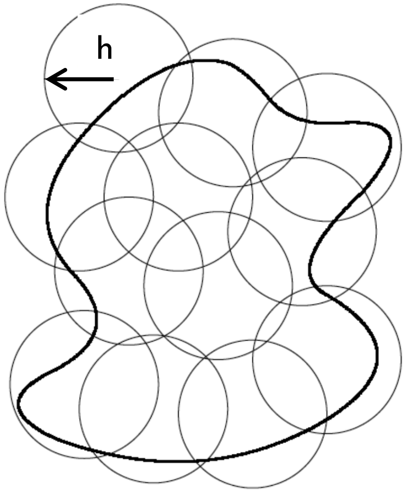
Scale to long sequences, $O(L \log L)$ instead of $O(L^2)$

# Usage in Machine/Deep Learning



- Q-learning with nearest neighbor [SX18]
  - Idea:
    - quantization of the state space X into $\{c_i\}_{i=1}^N$
    - (non-parametric) kernel ridge regression for new (x, a) values

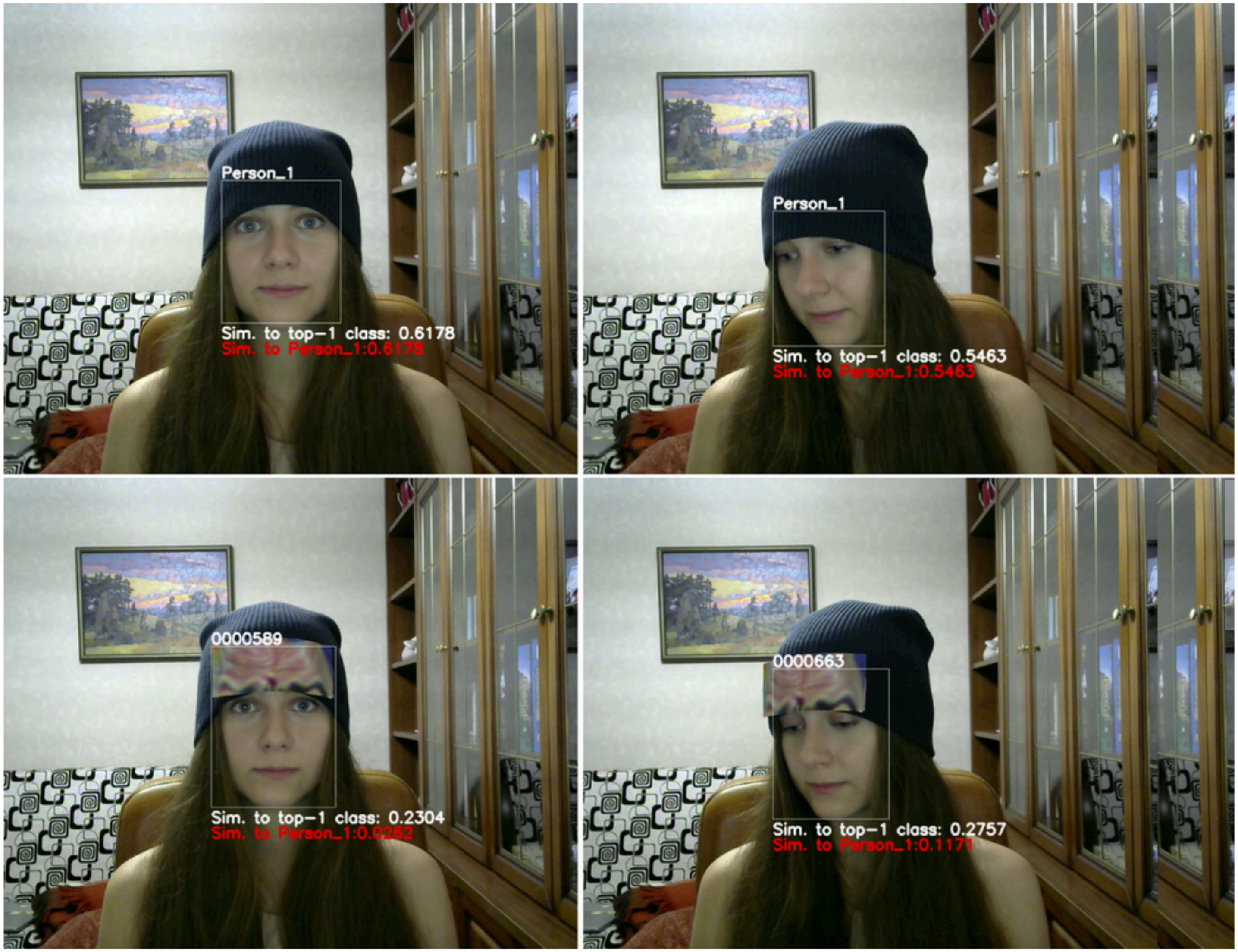
$$\hat{q}(x,a) = \sum_{i=1}^{n} K(x, c_i)\, q(c_i, a)$$
$$= \sum_{K(c,x) \le h} K(x, c)\, q(c, a)$$




Reinforcement Learning has been used in several DB problems, including Neo query optimizer [MNMZ+19]

# Example: Adversarial Machine Learning






sim ≥ 0.54

Adversarial sticker on the forehead

sim ≤ 0.28

[KP19]

# Example: Adversarial Machine Learning



- Local intrinsic dimensionality (LID) is an important feature to detect adversarial examples [MLWE+18]

$$\widehat{\mathrm{LID}}(x) = -\left(\frac{1}{k}\sum_{i=1}^{k}\log\frac{r_i(x)}{r_k(x)}\right)^{-1}$$

- High-dimensional geometry explains the existence of adversarial examples [GMFS+18]

require kNN queries

kNN queries are also useful in

- outlier/novelty detection
- kNN classification
- zero/few-shot learning

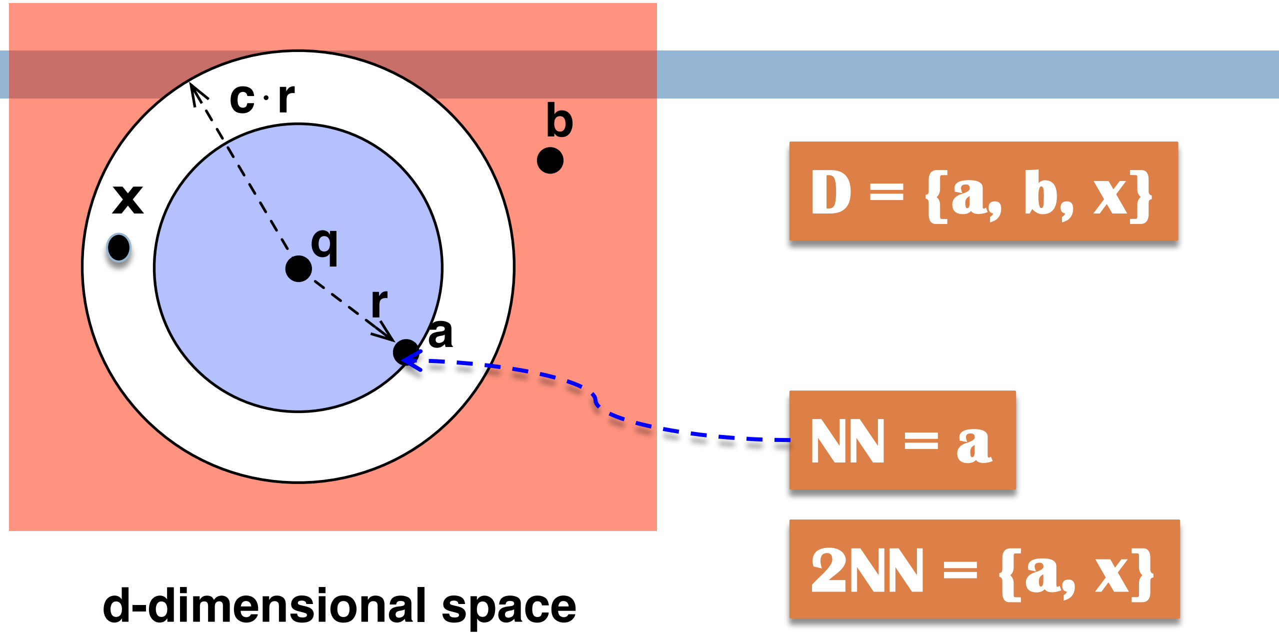
# Problem Definitions



- Database object & the query
  - d-dimensional point/vectors $\in R^d$
- Distance or similarity functions
  - dist(u, v) — $L_p$ distance ($0<p\leq 2$, $\infty$), Hamming dist, edit dist …
  - sim(u, v) — cosine similarity/inner product, Jaccard
- Query types
  - k-nearest neighbor queries (kNN)
  - range queries
  - conjunctive queries
  - similarity/distance join queries (top-k, range, closest pair, containment, …)

# NN and kNN



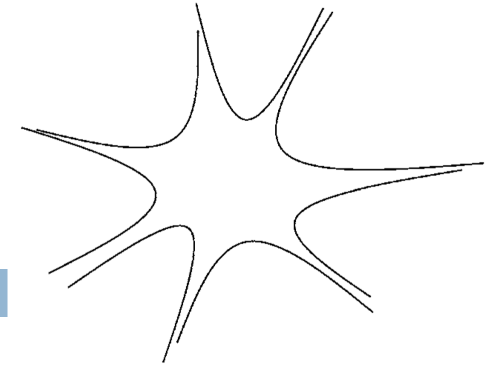



dist() is typically $L_2$ distance

- Nearest Neighbor (of q): o*
  - dist(o*, q) = min {dist(o, q), o ∈ D}
  - Generalizes to k-NN
- c-Approximate NN: o
  - dist(o, q) ≤ c * dist(o*, q)

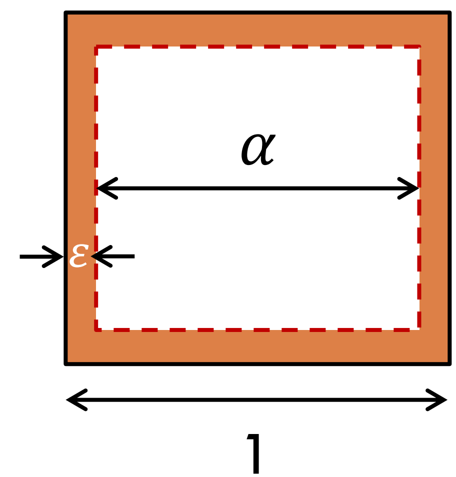
# Challenges /1

high-dimensional convex body





- Non-intuitive high-dimensional Geometry
  - Sampling uniformly within a unit hypercube ➔ samples are within a thin $\varepsilon$ 'shell'
    - Vol(r) = $\alpha^d \approx e^{-2\varepsilon d} \rightarrow 0$ ($\alpha < 1$)
  - Angle between two vectors
    - random Radamacher vectors ➔





$$\Pr\left[|\cos(\theta_{x,y})| > \sqrt{\frac{\log c}{d}}\right] < \frac{1}{c}$$

orthogonal w.h.p

Mohammed J. Zaki, Wagner Meira Jr. Data Mining and Analysis: Fundamental Concepts and Algorithms.

# Challenges /1



- Curse of Dimensionality / Concentration of Measure
  - Under some assumptions, maxdist(q, D)/mindist(q, D) converges to 1
    - Key assumption: independent distribution in each dimension
    - k-NN is still meaningful for real datasets

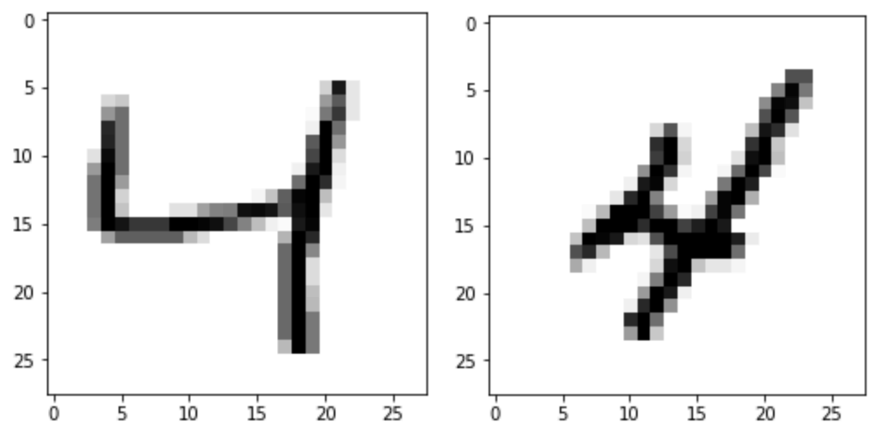
  - Hard to find algorithms sub-linear in *n* (# of points) and polynomial in d (# of dimensions)
  - Approximate version (c-ANN) is not much easier

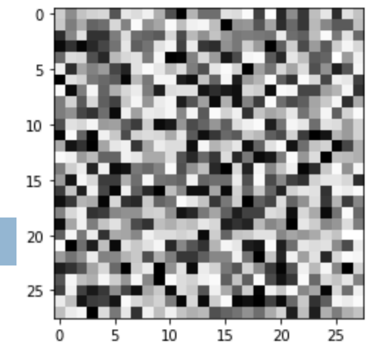
Asymptotic Geometric Analysis, Part I. Shiri Artstein-Avidan, Apostolos Giannopoulos, and Vitali Milman

# Challenges /2

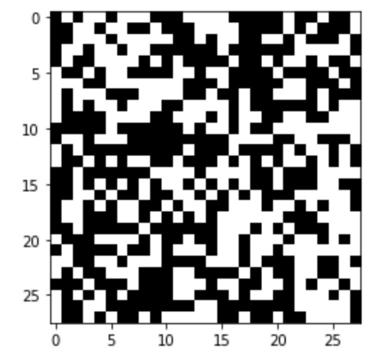


- No idea of the distribution of real data
  - Manifold hypothesis

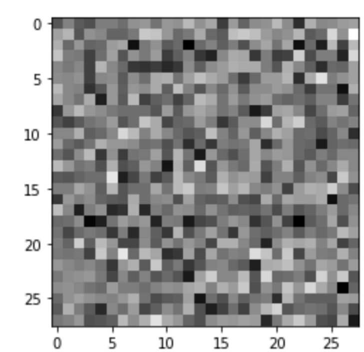




uniform



Radamacher



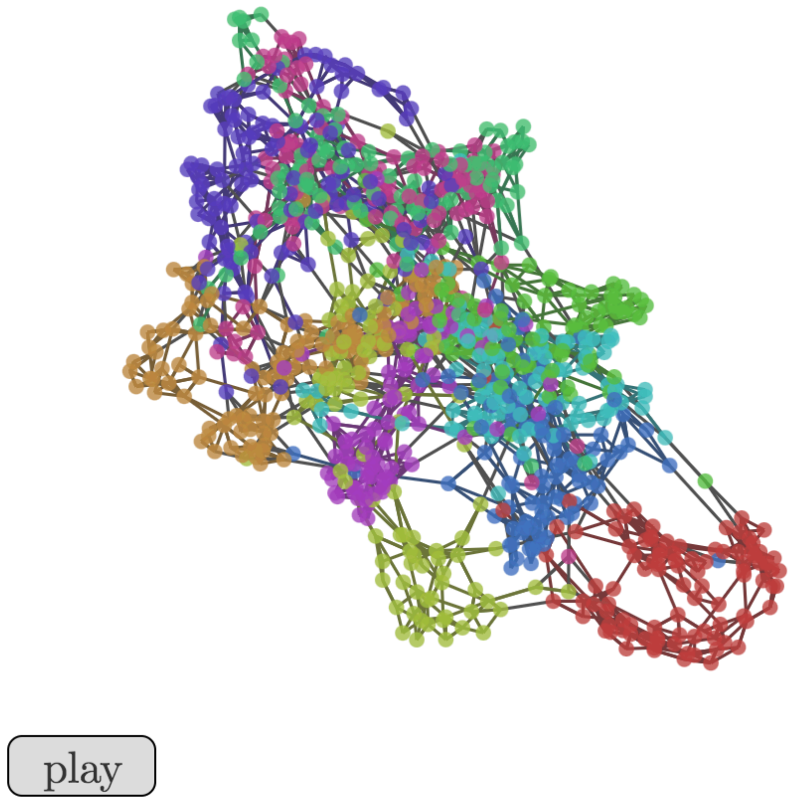
Gaussian

https://deepai.org/machine-learning-glossary-and-terms/manifold-hypothesis

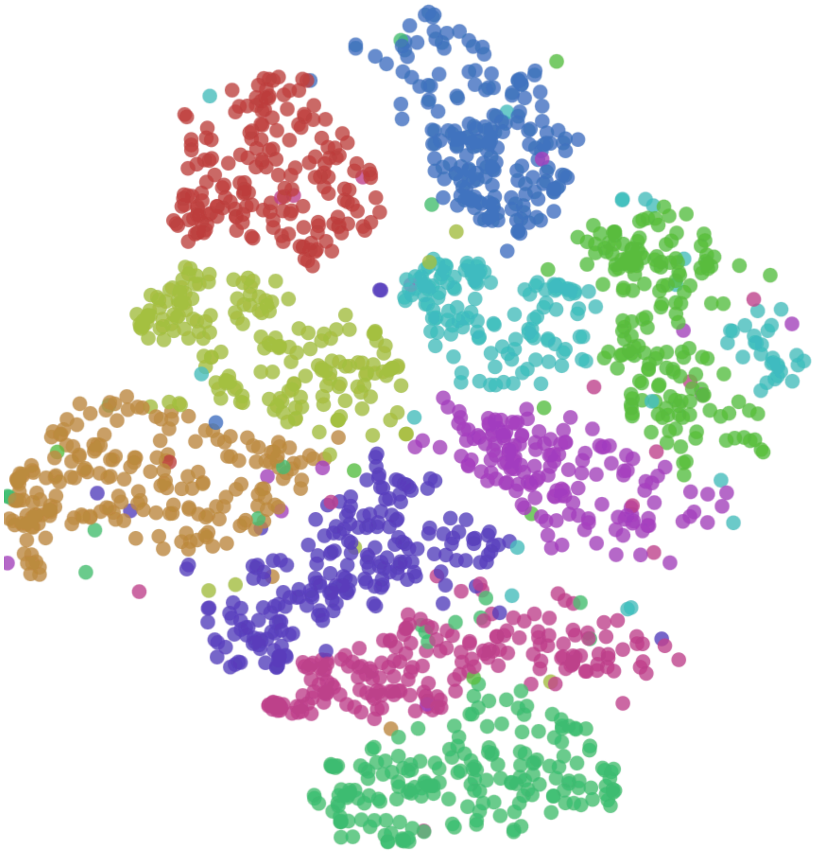
# Challenges /2



- Distribution of Real Data
  - Manifold hypothesis




Visualizing MNIST as a Graph



A t-SNE plot of MNIST

# Challenges /3



- Large data size
  - 1KB for a single point with 256 dims → 1B pts = 1TB
    - ~100 SIFT vectors per image
  - High-dimensionality (e.g., documents → millions of dimensions)
- Variety of distance/similarity functions
  - Less of an issue in the DL era