Dimension Reduction

07 April 2021 08:08

<u>Question</u> :-

- We have a collection of N images [1000 x 1000] pixels.
- We are given a "query" image. [1000 x 1000]
- Goal: quickly find out the "closest" image to the query image.

<u>How</u>?

<u>High level Idea</u>

- Transform each image into a vector in $1000^2 = 10^6$ dims.

$$\begin{array}{c} d \\ d \begin{bmatrix} 0 & .2 & .1 \\ .15 & .7 & .8 \\ 0 & .5 & .4 \end{bmatrix} \end{array} \rightarrow \overset{\longleftarrow D = d^2 \longrightarrow}{(0, .2, .1, .15, .7, .8, 0, .5, .4)}$$

- Same for query image
- Output the nearest vector to the query vector. (say l_2 distance)

(i.e) Given N vectors $v_1, v_2, \ldots, v_N \in \mathbb{R}^d$

$1^{\sim}$ and given query $q \in \mathbb{R}^d$

find $\boxed{\arg\min_{i=1}^{N} \|v_i - q\|_2^2}$

(APPROX.)
NEAREST NEIGHBOR SEARCH

Given N "base" vectors in D dimensions, pre-process and build a data structure.

So that when queries are presented, we can quickly retrieve the (approximate) closest vector to the query

---

<u>Naive Solution</u>

When query is presented, compare distances to all base points and output the closest.

operations per query : $3 \cdot N \cdot D + N$

D subtractions, Multiplications, Additions per point

find MIN.

In our example $D = 10^6$.

Today's lecture

generic "dimension reduction" technique which can bring down $10^6 \rightsquigarrow 10^3$ with very little "error".

JOHNSON - LINDENSTRAUSS LEMMA

Given n vectors in $\mathbb{R}^d$, $v_1, v_2, \ldots, v_n$ there is a mapping $f : \mathbb{R}^d \to \mathbb{R}^k$ such that

$\forall\ u_i, u_j$

$$(1-\varepsilon) \leq \frac{\|f(u_i) - f(u_j)\|_2}{\|u_i - u_j\|_2} \leq (1+\varepsilon)$$

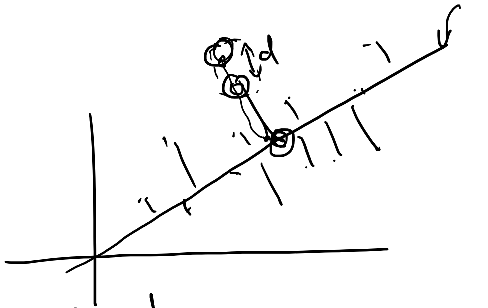
with $k = \Theta\left(\frac{\log n}{\varepsilon^2}\right)$.

Moreover we can find this efficiently with good probability.

Goal for today

Idea: PCA ?

- Unclear
- It's more of a
- global error reduction



- we want pairwise preservation

Idea 2: what about just picking k random coordinates?

$$v = (x_1, x_2, \dots, x_d) \quad \text{then} \quad f(v) = (x_{i_1}, x_{i_2}, \dots x_{i_k})$$

where $i_1, i_2, \dots i_k$ are randomly chosen from $[d]$.

Same $i_1, i_2, \dots i_k$ are used for all vectors.

[sample with replacement, to make analysis easy]

Good News

for any u and $v \in \mathbb{R}^d$

$$E\left[\, \|f(u) - f(v)\|^2 \right] = \frac{k}{d} \|u-v\|^2$$

good, b/c then we can think of

$$\hat{f}(u) = \sqrt{\frac{d}{k}} \left(u_{i_1}, u_{i_2}, \dots u_{i_k} \right)$$

and then we'll have

$$E\left[\, \|\hat{f}(u) - \hat{f}(v)\|^2 \right] = \|u-v\|^2.$$

Proof of Good News

Fix u, v and let $z = u - v$

$$\|z\|^2 = \sum_{i=1}^{d} (u_i - v_i)^2 = \sum_{i=1}^{d} z_i^2$$

Now, consider i_1 and focus on

$$\underset{i_1}{E}\left[\left(u_{i_1} - v_{i_1}\right)^2\right] = \sum_{l=1}^{d} \underbrace{\Pr[i_1 = l]}_{} \cdot (u_l - v_l)^2$$

$$= \frac{1}{d} \cdot \sum_{l=1}^{d} (u_l - v_l)^2$$

$$= \frac{1}{d} \|u - v\|_2^2$$

$$\Rightarrow \quad E\left[\|f(u) - f(v)\|_2^2\right] = \sum_{l=1}^{k} E\left[\left(u_{i_l} - v_{i_l}\right)^2\right]$$

$$= \frac{k}{d} \|u - v\|_2^2$$

If we can show some "concentration inequality" that $\|f(u) - f(v)\|$ is close to its average whp, then this scheme works!

and then union bound for all $1 \le i < j \le n$.

Q: Is this likely to have concentration?

A: Sadly, no ☹

Here's an example :-

$v_1 = (1, 0, 0, \ldots 0)$

$v_2 = (0, 1, 0, \ldots 0)$

$\vdots$

$v_n = (0, 0, \ldots, 1)$.

Unless we pick one of 1 or 2 coordinate in f,

$\|f(u_1) - f(u_2)\| = 0$

$\Rightarrow$ even if $k < n-1$, we will always make huge error on some pair.

real distance $= \sqrt{2}$

"our imagined" distance $= 0$

badness because most coordinates are same, few coords give all the distance.

How to "spread" this over all coords?

TRICK:

- Nothing sacred about standard basis.

choose a random rotation $R \in \mathbb{R}^{d \times d}$

first rotate the data

$$\boxed{\begin{array}{l} V_i \longrightarrow RV_i \quad (\text{still in } d\text{-dim}) \\ \text{then sample } k \text{ coords from } RV_i \end{array}}$$

- Intuition, can be made rigorous
- can also be simplified greatly using gaussians.

$$\boxed{f(u) = \frac{1}{k} \underbrace{\overset{d}{\begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1d} \\ \vdots & & & \\ g_{k1} & g_{k2} & \cdots & g_{kd} \end{bmatrix}}}_{G} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u \end{bmatrix}}$$

09/04/2021

$$f(u) = \lambda \; \begin{array}{c} \uparrow \\ k \\ \downarrow \end{array} \overset{\longleftarrow d \longrightarrow}{\begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1d} \\ g_{21} & \cdots & & g_{2d} \\ \vdots & & & \\ g_{k1} & & \cdots & g_{kd} \end{bmatrix}} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ \vdots \\ u_d \end{bmatrix}$$

$G =$ random $k \times d$ gaussian matrix where each g_{ij} is an independant gaussian r.v with mean 0 & variance 1.

i.e. $g_{ij} \sim N(0, 1)$.

λ will be chosen suitably later.

<u>THEOREM</u>

For suitable λ and k, this mapping preserves <u>all</u> pairs of distances upto $(1 \pm \varepsilon)$ factor.

<u>Q</u> How do we prove this?

We'll fix a pair of vectors u_i, u_j and let $z = u_i - u_j$

Next we show that

$$\Pr_{\substack{\text{random choice} \\ \text{of } G}}\left[\frac{\|f(u_i) - f(u_j)\|_2^2}{\|u_i - u_j\|_2^2} \notin (1-\varepsilon, 1+\varepsilon)\right] \leq \frac{1}{n^2}$$

Then we just do union bound over all pairs u_i, u_j.

$$\Pr_G \left[\exists\, i,j \text{ which is not preserved to } (1 \pm \varepsilon) \right] \leq \frac{\binom{n}{2}}{n^2} \leq \frac{1}{2}.$$

With prob $\frac{1}{2}$, experiment was successful (ie) all pairs preserved!

If not, just repeat ☺

Need to show that a single pair is preserved w.h.p.

Fix u & v and let $Z = u - v$

$$f(u) = \lambda \cdot G \cdot u$$
$$f(v) = \lambda \cdot G \cdot v$$

$$f(u) - f(v) = \lambda \cdot G (u - v) = \lambda G Z = f(Z).$$

Need to show that

$$1 - \varepsilon \leq \frac{\|f(Z)\|_2^2}{\|Z\|_2^2} \leq 1 + \varepsilon \quad \text{with prob} \geq 1 - \frac{1}{n^2}$$

let $Z = (Z_1, Z_2, \ldots, Z_d)$.

and consider $\hat{Z} = \frac{Z}{\|Z\|}$.

Then $Z = \|Z\| \cdot \hat{Z}$.

Moreover,

$$f(z) = \lambda \cdot G \circ z = \lambda \|z\| \cdot G \hat{z}$$
$$= \|z\| \cdot f(\hat{z}).$$

So, it suffices to show that

$$(1-\varepsilon) \le \|f(\hat{z})\|_2^2 \le (1+\varepsilon) \quad \text{w.p.} \ge 1 - \frac{1}{n^2} \quad (*)$$

$$\Rightarrow \quad \|z\|^2 (1-\varepsilon) \le \|f(z)\|_2^2 \le (1+\varepsilon)\|z\|^2$$

Consider $\hat{z} = (\hat{z}_1, \hat{z}_2, \dots, \hat{z}_d)$

We know $\|\hat{z}\|_2^2 = 1 \Rightarrow \sum \hat{z}_i^2 = 1$.

Let's look at $f(\hat{z}) = \lambda \cdot G \circ \hat{z}$

$$= \lambda \begin{bmatrix} \sum_{j=1}^{d} g_{1j} \hat{z}_j \\ \sum_{j=1}^{d} g_{2j} \hat{z}_j \\ \vdots \\ \sum_{j=1}^{d} g_{kj} \hat{z}_j \end{bmatrix}$$

Each entry in $f(\hat{z})$ looks like

$$\lambda \cdot N\left(0, \sum_{j=1}^{d} \hat{z}_j^2\right) = \lambda N(0,1).$$

Moreover because g_{ij} and $g_{i'j'}$ are independent for all independent for all i, j, i', j',

these entries are themselves independent!

$$f(\hat{z}) = \lambda \begin{pmatrix} F_1 \\ F_2 \\ \vdots \\ F_k \end{pmatrix} \quad \text{where } F_i \sim N(0,1).$$

What is $E\left[\|f(\hat{z})\|^2\right] = \lambda \sum_{i=1}^{k} E\left[F_i\right]^2 = \lambda \cdot k.$

We want $f(\hat{z})$ to be close to 1.

$\Rightarrow$ set $\boxed{\lambda = \frac{1}{k}}$

$$\Rightarrow E\left[\underbrace{\|f(\hat{z})\|^2}\right] = 1.$$

Let $Y = k\|f(\hat{z})\|^2$ be the random variable.

$$Y = \underbrace{\sum_{i=1}^{k} F_i^2}$$

Y is the sum of many independent random variables.

Moreover each F_i is a gaussian $N(0,1)$

$\boxed{\chi^2\text{-distribution}}$

χ^2 - distribution

{ Y actually sharply concentrates around its mean. }

Try to prove this using ideas from chernoff bounds proof

More or less,

Y behaves like $N(\mu(Y), \sigma^2(Y))$

$$E[Y] = \sum E[F_i^2] = k.$$

$$Var[Y] = \sum_{i=1}^{k} Var[\underbrace{F_i^2}]$$

$$Var[F_i^2] = E[F_i^4] - E[F_i^2]^2$$

$$= \underbrace{E[F_i^4]} - 1.$$

{ $E[F_i^4] = O(1)$ [constant] for gaussian distribution.

$\Rightarrow Var[Y] = Ck$ for constant C.

{ Y behaves like $N\underbrace{(k, Ck)}$. }

Very big cheat, crude approx.

Basically, we'll use chernoff-like tail bounds on Y.

tail bounds on Y.

$$\Pr\left[\,|Y - E[Y]| \geq t\,\right]$$

$$\leq \exp\left(\frac{-t^2}{\sigma^2}\right)$$

Can prove this type of inequality for Y.

$E[Y] = k$

$\sigma^2(Y) = C \cdot k$.

Set $t = \varepsilon \cdot k = \varepsilon E[Y]$

$$\Pr\left[\,Y \notin (1-\varepsilon, 1+\varepsilon) E[Y]\right] \leq \exp\left(\frac{-\varepsilon^2 k^{\cancel{2}}}{C \cdot \cancel{k}}\right)$$

$$= \exp\left(\frac{-\varepsilon^2 k}{C}\right).$$

So just set $\boxed{k = \frac{2C \cdot \ln n}{\varepsilon^2}}$

$$\Pr(\text{bad event}) \leq \exp\left(\frac{-\cancel{\varepsilon^2} \cdot 2\cancel{C} \ln n}{\cancel{C}\,\cancel{\varepsilon^2}}\right)$$

$$= \frac{1}{n^2}.$$

Whenever Y lies $\in [1-\varepsilon, 1+\varepsilon]\, E(Y)$

$\iff f(z) = \frac{1}{k} Y$ lies in $[1-\varepsilon, 1+\varepsilon]$.

$\Leftrightarrow \quad f(\hat{z}) = \frac{1}{k} \cdot Y$ lies in $[1-\varepsilon, 1+\varepsilon]$.

$\Rightarrow \Pr\left(f(\hat{z}) \notin [1-\varepsilon, 1+\varepsilon] \right) \leq \frac{1}{n^2}$ for

$k = \frac{2c \ln n}{\varepsilon^2}$

and $\lambda = \frac{1}{k}$.

Locality Sensitive Hashing

12 April 2021 10:00

(c, r) - Approximate Near Neighbor search.

Motivation

- Again vectors in high dimensional space
- look to retrieve "closest" vector to given query vector.

Concrete Example :-

N documents, N is very large (all in english)

Represented as a $\{0,1\}^D$ vector for D very large.

$D =$ all distinct english words Ordered as $\{w_1, w_2, \ldots w_D\}$

$v = (v_1, v_2, \ldots, v_D)$

where $v_i = 0$ if i^{th} word is not in document

$= 1$ if i^{th} word is in doc.

Goal: build a data structure / Algorithm s.t when query vector q arrives, we quickly retrieve any nearby base vector.

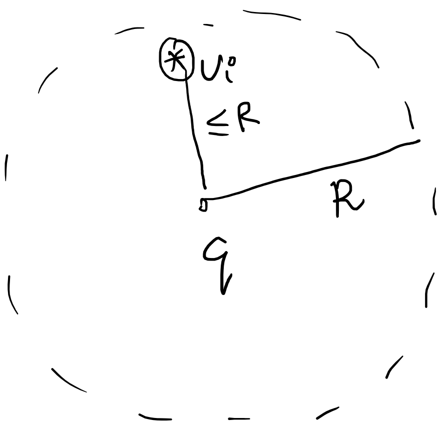
$q = (q_1, q_2, \ldots, q_D)$

Given N base vectors $v_1, v_2, \ldots, v_N$ and $q = (q_1, q_2, \ldots, q_D)$ arrives, and given parameter R, find any base vector which are at distance $\leq R$ from query q, if they exist.

Distance = Hamming distance

$$= |u - v|_1 = \sum_{j=1}^{D} |u_i - v_i|$$

$$= \#\text{indices where they differ}$$



<u>Naive approach</u>

- Compare distances to all base vectors
Output the closest.

Query Time = $O(\underline{N}D)$

Space of data structure = ND bits
(store all base vectors).

<u>QN</u>:

Can we get sub-linear dependence on N?
eg, even if we bring query time
from ND to $\sqrt{N} \cdot D$, that is
really significant in real life
N can be 10^9, 10^{12} even!!

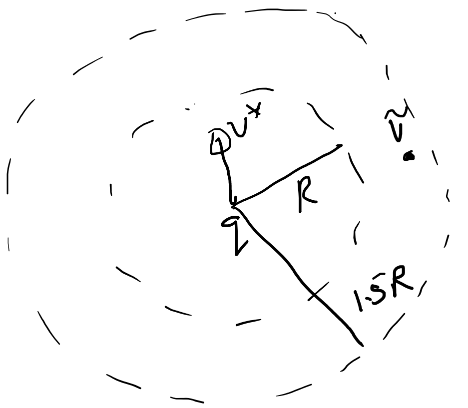
Today + Tomorrow:

Can do such a thing if we're OK
with <u>Approximate</u> Near Neighbor search

Given q, target R,

- Algo can output <u>NO</u> if no base
point is distance $\leq R$
from q.

- Output **any** base point at distance $\leq CR$ from q, for some $C > 1$.



[Algo can output $\tilde{v}$ instead of v^*]

In this case $C = 1.5$

C is fixed up-front, is like "approximation factor of algo".

THM [Indyk-Motwani '90s]

(*) For above problem, can design C-Approx NNS algorithm with space $O(N^{1+1/c} + ND)$ but query time $= O(N^{1/c} D)$.

Idea: Design very generic technique called "LOCALITY SENSITIVE HASHING".

Imagine the following type of search Algo:-

Compute a hash function

Compute a hash function

$h: \{0,1\}^D \to [K]$ eg. where $K \ll N$

think of $K = \sqrt{N}$.

It maps each document vector to one of K buckets.

Similarly it can map query to a bucket.

Algo: Only look in the query bucket and output the closest base vector.

Query time: $\approx \frac{N}{K} \cdot D \approx \sqrt{N} \cdot D$ if $K = \sqrt{N}$.

h is a random hash $\Rightarrow$ each bucket has $\approx N/K$ pts.

Issue

No reason why query's nearest neighbor is hashed to same bucket as q.

Idea:

What if hash fn 'h' is locality sensitive?

(ie) nearby points are in same bucket more often than far away points!

than far away points!

Then Maybe such an algo would work!

LSH

A hash family $\mathcal{H} = \{h_1, \dots h_M\}$ is a (C, R, P_1, P_2) LSH for $P_1 > P_2$ if

for any 2 vectors x and y,

① $\Pr_{h \sim \mathcal{H}} [h(x) = h(y)] \geq P_1$ if $d(x,y) \leq R$

② $\Pr_{h \sim \mathcal{H}} [h(x) = h(y)] \leq P_2$ if $d(x,y) > CR$

Property ② ensures that buckets are small on avg, and

① ensures that near neighbors of query are in same bucket as q.

$$\boxed{\{\text{From } (C, R, P_1, P_2) \text{ LSH to } (C, R) \text{ Algo.}\}}$$

Idea:

① "boost" the gap between P_1 and P_2 by taking

① by taking

$$g(x) = [h_1(x), h_2(x), \dots h_k(x)]$$

where $h_1, h_2, \dots h_k$ are k independent samples from $\mathcal{H}$.

What does g satisfy:

$\Pr[g(x) = g(y)] \geq p_1^k$ if $d(x,y) \leq R$

$\Pr[g(x) = g(y)] \leq p_2^k$ if $d(x,y) > CR$

- This has driven down p_2^k ⇊
- Now, this also has driven down good case collision (p_1^k)
- Fix this by using L different hash functions $g_1, g_2, \dots, g_L$ {independently for each g_i}

<u>Overall Algo</u> :-

① Compute $g_i = (h_{i1}, h_{i2}, \dots h_{ik})$ for $i = 1, 2, \dots L$.

② ... and

② Compute $g_i(v)$ for all $1 \le i \le L$ and all base vectors.

When query arrives

Compute $g_1(q), \ldots, g_L(q)$.

For $i = 1, 2, \ldots L$

- Look at all base vectors with $g_i(v) = g_i(q)$.
- If we find any at distance $\le cR$, output and terminate

Need to analyze time and space complexity.

Space Complexity: $N \cdot D + \underbrace{O(N \cdot K \cdot L)}_{\text{hash values}}$

Query Time :-

$LK + \underbrace{L}_{\#\text{ hash fns } g} * \left(E[\#\text{ far away pts which fall into bucket of } q] \cdot \underbrace{D}_{\text{dimension}} \right)$

sort of "wasteful computation"

sort of
"wasteful computation"

$$\leq L \cdot N \cdot P_2^k \cdot D$$

$$E[\text{query time}] \leq L N P_2^k \cdot D$$

Now its all about parameter choosing

Choose K ~~st~~ $P_2^k = \frac{1}{N}$.

(ie)

$$K \log \frac{1}{P_2} = \log N$$

$$K = \frac{\log N}{\log 1/P_2}.$$

Next choose L so that q's near-neighbor is in the bucket with good probability

Let v^* be q's neighbor within distance $\leq R$ (if any).

What is the probability that it falls into one of the L buckets of q?

Pr [v^* falls in q's bucket in one of the L hash fns]

$$\geq 1 - \prod_{i=1}^{L} (1 - p_1^k)$$

$$= 1 - (1 - p_1^k)^L$$

$(1-\varepsilon)^L$

$\approx 1 - \varepsilon L$

when ε is small

Need to formalize $\rightarrow \approx \quad L \cdot p_1^k$

Set L st

$$L \cdot p_1^k = 1$$

to ensure good success probability

$$L = \frac{1}{p_1^k}$$

$$= \left(\frac{1}{p_1}\right)^{\frac{\log N}{\log(1/p_2)}}$$

$$\boxed{L = N^{\frac{\log\left(\frac{1}{p_1}\right)}{\log\left(\frac{1}{p_2}\right)}}} \leftarrow \rho$$

Parameter $\rho = \dfrac{\log(1/p_1)}{} < 1$ is

$\log(1/p_2)$
the important parameter of LSH.

Summary

$$k = \frac{\log N}{\log 1/p_2} \; ; \; L = N^\rho$$

Expected Query Time

$$= kL + O(N \cdot \underbrace{L \cdot p_2^k}_{} \cdot D)$$

$$= \frac{\log N}{\log 1/p_2} \cdot N^\rho + O(N^\rho \cdot D) .$$

{great if $\rho < 1$} ☺

Success Prob

if $\exists$ a near nbr, we find some approx NN with prob

$$\geq \left(1 - (1 - p_1^k)^L\right)$$

$$\geq 1 - e^{-p_1^k \cdot L}$$

$(e^x \geq 1 + x)$

$$\geq (1 - \ ) .$$ ∎

$(e^x \geq 1+x)$ $\geq (1 - 1/e)$. ∎

THM :

(C, R, P_1, P_2) LSH with $\rho = \dfrac{\log 1/P_1}{\log 1/P_2}$

implies a randomized (C, R) - ANNS algorithm with success prob $\geq (1 - 1/e)$ of finding a near nbr @ distance $\leq CR$ (if there exists some nbr at dist $\leq R$) and expected query processing time

$$= O\left(N^{\rho} \cdot \left(\frac{\log N}{\log 1/P_2} + D\right)\right)$$

$\leftarrow$ constant, can ignore

$\Rightarrow$ LSH type hash fns are good as long as ρ is small !! :)

QN

for vectors in $\{0,1\}^D$ with $d(x,y) = \sum_{i=1}^{D} |x_i - y_i|_1$ as distance, What is a good LSH ?

Want

- If x, y are close by ($\leq R$)
  $\Pr[h(x) = h(y)]$ is large.
- If x, y are far apart ($\geq CR$)
  $\Pr[h(x) = h(y)]$ is small.

Dirt Simple Idea

Pick a random coordinate d from $\{1, 2, \ldots, D\}$.

$$\boxed{h_d(x) = x_d}$$

$$\mathcal{H} = \{h_1, h_2, \ldots h_D\}$$

P_1, P_2 ANALYSIS

If $d(x,y) \leq R$, (ie) they differ in at most R places,

$$\Pr_d[h_d(x) = h_d(y)] \geq \underbrace{\left(1 - \frac{R}{D}\right)}_{P_1}.$$

$$\Pr_d[h_d(x) = h_d(y)] \leq \underbrace{\left(1 - \frac{CR}{D}\right)}_{P_2}.$$

$\underbrace{\qquad}_{P_2}$

if $d(x,y) \geq cR$

What is the ρ of this hash fn.

$$\rho = \frac{\log\left(1/P_1\right)}{\log\left(1/P_2\right)}$$

$$= \frac{\log\left(1 \big/ \left(1-\frac{R}{D}\right)\right)}{\log\left(1 \big/ \left(1-\frac{cR}{D}\right)\right)}$$

Need to be formalized $\Bigg\{$ Assuming $R \ll D$, we'll make some simplifications

a) $\frac{1}{1-\varepsilon} \approx 1+\varepsilon$

b) $\log(1+\varepsilon) \approx \varepsilon$

$\Bigg\}$ if $\varepsilon \ll 1$.

<u>Hence</u> $\rho \approx \frac{R\cancel{D}}{\cancel{D} \cdot cR} = \frac{1}{c}$.

Using this hash family gives us the (C, R) - Near Neighbor search problem with

space $O\left(N^{1+1/C} + ND\right)$ and

Query time $O\left(N^{1/C} \cdot (D + \log N)\right)$.

$\Downarrow$

· Represents a significant milestone for "Nearest vector search" and is used in variety of applications today !!