Advanced Algorithms

Lecture 16: High Dimensional Geometry and the Curse of Dimensionality

Lecturer: Nathan Klein

1 High Dimensional Geometry and the Curse of Dimensionality

Remember we discussed in the previous lecture that the nearest neighbor search problem in 1-dimension can be solved in time $\log(n)$ by sorting the entries and using binary search. If you're careful, you can make this work in more dimensions too, but you will quickly notice a bad dependence on d of around 2^d. In the first part of this lecture we will formalize why this happens.

To make the issue concrete, suppose we have a bunch of points in $\mathbb{Z}^2$. Now imagine you want to solve the radius-constrained approximate nearest neighbor problem on this set using the ℓ_1 norm. Let's make the problem even easier and ask: **is there a point in our set at distance exactly r from a point (x, y)?**

- In $\mathbb{R}^2$, this can be done in time $O(r)$. We can try all of the possible points at distance r and see if they are in the set. In particular, this is $(x \pm r, y), (x \pm (r-1), y \pm 1), \ldots, (x, y \pm r)$. There are $r+1$ elements in this series. For the first and last in the series, there are two possible points, and for the remainder, four possible points, giving a total of $4(r-1) + 2 \cdot 2 = 4r$. Formally, what am I doing? I am starting at point (x, y) and then checking $(x, y) + \alpha$ for all points $\alpha \in \mathbb{Z}^2$ with $\|\alpha\|_1 = r$.
- Now what about in $\mathbb{Z}^d$? Well, let's do the same thing: start at $x \in \mathbb{Z}^d$ and then check if $x + \alpha$ is in the set for all points $\alpha \in \mathbb{Z}^d$ with $\|\alpha\|_1 = r$. It turns out that when d is big, there are a huge number of such points. This is like looking at a discrete ℓ_1 sphere of radius r in dimension d and counting the number of points on it.

Lemma 1.1. *The number of points on the ℓ_1 sphere with radius r in $\mathbb{Z}^d$, assuming $r \geq d$, is at least $2^{d-1} \frac{(r-1)^d}{(d-1)^{d-1}}$. For $r \gg d$ this is approximately $2^d r^{d-1}$.*

Proof. Rephrased, we want find the number of ways to pick numbers $r_1, \ldots, r_d \in \mathbb{Z}$ with $\sum_{i=1}^d |r_i| = r$.

We can lower bound this using a classic "stars and bars" counting problem. We have r stars in a line. Now we want to pick the position of $d+1$ "bars" $1, \ldots, d+1$, where the first bar is fixed at the left side of the r stars and the last bar at the right side. In this way, r_i is equal to the number of stars between the ith bar and the $i+1$st. This will allow us to count assignments where all r_i are positive, which is a lower bound on the number of points in the boundary.

So really what's going on is we have a string of $r + d - 1$ characters (the first and last are fixed) and we just need to pick the location of the $d-1$ bars. This is then exactly $\binom{r+d-1}{d-1}$ which for fixed d grows like $(r+d-1)^{d-1}$.

To more accurately account for the asymptotics of the number of points on the boundary when $r \gg d$, we can do as follows: constrain that every r_i is at least 1, and multiply everything by 2^d since we can pick each to be positive or negative. To do this, I can just throw out d stars, assigning

one to each r_i, and then repeat the argument so that I have $r + d - 1 - d = r - 1$ characters giving a bound of:

$$2^d \binom{r-1}{d-1} \geq 2^d \frac{(r-1)^{d-1}}{(d-1)^{d-1}} \approx 2^d r^{d-1}$$ □

In other words, the number of possible points grow *exponentially* with d. This is why using approximation and schemes like LSH are crucial to avoiding running time of $O(n)$ for nearest neighbor search.

1.1 A Geometric View

The counting argument gives a nice combinatorial point of entry into the curse of dimensionality, but to get a broader sense of what's going on it can be useful to look at the geometry in $\mathbb{R}^d$. Let's check out high dimensional balls. Remember that a ball with radius r is the collection of points $x \in \mathbb{R}^d$ with ℓ_2 distance at most r from the origin, i.e., $\sqrt{\sum_{i=1}^d x_i^2} \leq r$.

Fact 1.2. *The volume of a ball with radius r in $\mathbb{R}^d$ is $\frac{\pi^{d/2}}{(\frac{d}{2}-1)!} r^d$. (For d odd, this is a bit funky, satisfying $(\frac{d}{2}-1)! = \frac{d}{2} \cdot (\frac{d}{2}-1) \cdot \cdots \cdot (\frac{1}{2}) \cdot \pi^{1/2}$.)*

Just for a sanity check, in two dimensions, we get back πr^2 using that $0! = 1$. In three dimensions, we get $\frac{\pi^{3/2}}{\frac{3}{2} \cdot \frac{1}{2} \cdot \pi^{1/2}} r^3 = \frac{4}{3}\pi r^3$.

This immediately gives us something quite interesting. Let B_r^d be the ball of radius r in dimension d.

Fact 1.3. *Sample a uniformly random point x in a ball of radius r in dimension d. The probability that the distance between the origin and x is at most $(1-\epsilon)r$ goes to 0 as $d \to \infty$.*

Proof. The probability x lies in the ball of radius $(1-\epsilon)r$ is exactly:

$$\frac{\text{Volume of } B_{(1-\epsilon)r}^d}{\text{Volume of } B_r^d} = \frac{r^d}{((1-\epsilon)r)^d} = \frac{1}{(1-\epsilon)^d}$$

since we sample a uniformly random point from B_r^d and x lies in $B_{(1-\epsilon)r}^d$ if its distance from the origin is at most $(1-\epsilon)r$, which goes to 0 as $d \to \infty$. □

OK that's a bit weird. It means that as d grows, almost none of my points are near the origin: 99.9% of them lie in a little slice of the ball, between the $(1-\epsilon)r$ ball and the r ball.

Maybe the origin is special then. Probably it's at least true that if I sample a couple of uniformly random points in the ball B_r^d, they are at least fairly close to each other, say about $r/2$, right?

Wrong.

Lemma 1.4. *Sample two points a, b from B_r^d uniformly at random. Then the probability the distance between a and b is at most $(1-\epsilon)r$ again goes to 0 as $d \to \infty$.*

Proof. Fix point a. Now, for b to be within a radius of $(1-\epsilon)r$ of a, we need to pick b in the $(1-\epsilon)r$ ball centered at a. Part of this ball lies within B_r^d, but the more of it lies in B_d^r, the higher the probability b is within distance $(1-\epsilon)r$ of a. So to upper bound the probability a and b are close, we can assume it all lies in B_r^d. But now this is the same calculation as before, and we get $\frac{1}{(1-\epsilon)^d}$ which goes to 0. □

And in fact, with high probability their distance will be larger than r (which you will prove on the homework). This is bizarre to visualize, but it says that pretty much every pair of points we sample from B_r^d (for d large) is far away.

Another Facet of the Curse. Forgetting about the runtime of nearest neighbor search, you might imagine that you would like to build a large dataset so that *most query points are close to some point in your dataset*. After all, isn't that the point of a nearest neighbor query? In the eBay example, you would like to find a product that's actually related to what someone is searching for; in the movie example, you'd like to recommend a movie that is similar to someone's favorite film.

But what the above calculations tell us is that if d is large compared to n, your number of points, and your points are sampled from B_r^d, that's not going to happen. Until you get exponentially in d many points, all of your points will be at distance at least r from each other, swimming alone in the void of high dimensional space.

A famous example of this came up in the design of cockpits for aircraft around the 1940s and 50s. Designers took a measurements of a sample of people: their heights, weights, arm lengths, leg lengths, and so on, and then took the average of *each* statistic to determine the measurements of the cockpit, which needs to be calibrated to these numbers. In some NASA reports, this was called the "percentile man" approach.

It didn't work. To quote the report:

> *There are a number of errors inherent in the "percentile man" approach which have resulted in marked difficulties for a number of pilots operator or escaping from their aircraft.*

Reading between the lines (difficulties escaping), this must have resulted in the death or serious injury of some pilots. Could we have predicted that this was a bad idea?

I would say, almost certainly, yes. Let each person be represented by a d-dimensional vector $x \in \mathbb{R}^d$. Scale things so that the "percentile man" lives at the origin. Now, imagine that a person with measurements x is sampled from B_r^d: for some r this is not an unreasonable model since the expected value of each coordinate is 0 and we would expect some deviation.

But the corollary of Fact 1.3 is that with high probability, *percentile man does not exist*, or (less hyperbolically), your typical pilot is highly unlikely to be close to percentile man. Taking the average of many dimensions does not at all ensure you will be able to design a cockpit that fits most pilots. Large deviation from the origin is actually the expected outcome in high dimensions.

1.2 Dimension Reduction

Next class we will discuss an important result about high dimensional spaces called the Johnson-Lindenstrauss lemma. Given n points $x_1, \ldots, x_n \in \mathbb{R}^d$ and an approximation factor $\epsilon > 0$, we will show that it is possible to find a collection of n points $y_1, \ldots, y_n \in \mathbb{R}^k$ for $k \in O(\frac{\log n}{\epsilon^2})$ so that the ℓ_2 distances between all pairs of points are approximately preserved after mapping each point x_i to y_i. In other words, for all i, j we have:

$$(1-\epsilon)\|x_i - x_j\|_2 \leq \|y_i - y_j\|_2 \leq (1+\epsilon)\|x_i - x_j\|_2$$

This is a very useful fact. Suppose you have a collection of n vectors $S \subseteq \mathbb{R}^d$ for some very large d, and given any $x, y \in S$ you want to be able to compute their distance $d(x, y) = \|x - y\|_2$. This

lemma says that you can reduce the storage from nd to $O(n \log(n))$ as long as you are OK with a small loss in precision.

The algorithm will be very simple: let G be a matrix in $\mathbb{R}^{n \times d}$ where each entry $G_{i,j}$ is a random independent Gaussian with mean 0 and variance 1, $\mathcal{N}(0,1)$. Then we will let $y_i = \frac{1}{\sqrt{n}} G x_i$ for all i. Next time we will do the analysis.