CSE 632 (Fall 2019): Analysis of Algorithms II : Randomized Algorithms

Lecture 13-14 (10/9/2019, 10/11/2019): Johnson Lindenstrauss

Lecturer: Shi Li *Scriber: Hanping Zhang, John Rivera*

1 Dimension Reduction

- Problem: Given n data points in a d-dimensional space, where d may be very large.
- Fact: Many algorithms are inefficient (in terms of time or space) if d is big.
- Goal: We want to reduce the dimension d. Meanwhile, we maintain the pointwise distances between any 2 of the n points.

Theorem 1. *Suppose $a_1, a_2, ..., a_n$ are n points in R^d and $\epsilon \in (0,1)$. For some $k = O(\frac{\log n}{\varepsilon^2})$, there exists a linear function $f : R^d \to R^k$ such that:*

$$\forall i, j : (1-\varepsilon)\|a_i - a_j\|^2 \leq \|f(a_i) - f(a_j)\|^2 \leq (1+\varepsilon)\|a_i - a_j\|^2.$$

The idea is of the proof is that we randomly project the n points to a k-dimensional space. In order to show how to do a random projection, we need to use normal distributions.

2 Normal distribution (Gausian distribution)

The standard normal distribution $\mathcal{N}(0,1)$ has the probability density function $f(x) = \frac{1}{\sqrt{2\pi}}e^{-\frac{x^2}{2}}$.

Properties of the standard normal distribution: (Below, assume X is a variable following the distribution.)

- symmetric around 0
- highly concentrated
  - $\Pr[X \in [-1,1]] \approx 0.683$
  - $\Pr[X \in [-2,2]] \approx 0.955$
  - $\Pr[X \in [-3,3]] \approx 0.997$
- expection (mean) = 0, standard deviation = 1
- The Cumulative Distribution Function (CDF)

$$\Phi(x) = \Pr[X \leq x] = \int_{-\infty}^{x} f(t)dt$$

  has no closed form.

In general, we have the normal distribution $\mathcal{N}(\mu, \sigma^2)$ with mean μ and variance σ^2.

- Probability Density Function (PDF) of $\mathcal{N}(\mu, \sigma^2)$:

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- Suppose $X_1, X_2, ..., X_n$ are n independent variables following the standard normal distribution, and $\alpha_1, \alpha_2, ..., \alpha_n \in \mathbb{R}$. Then,

$$\sum_{i=1}^{n} \alpha_i X_i \sim \mathcal{N}\left(0, \sum_{i=1}^{n} \alpha_i^2\right).$$

With normal distribution, we can choose a direction uniformly at random. First consider the cases in 2-dimension.

- If we choose $x, y \in [-1, 1]$ independently and uniformly at random and use (x, y) as the direction, this does not give a uniformly random direction. The up-right direction $(1, 1)$ will have higher density than the right direction $(1, 0)$.
- We can choose uniformly at random $\theta \sim [0, 2\pi)$ and let $(x = \cos\theta, y = \sin\theta)$. Then this is a uniform direction. However this approach is hard to be extended to the 3-dimensional case.

Indeed, we can choose $x, y \sim \mathcal{N}(0, 1)$ independently. Then, $\frac{(x,y)}{\sqrt{x^2+y^2}}$ is a direction uniformly at random chosen. This can be generalized to d dimensions: $X_1, X_2, ..., X_d$ randomly and independently from $\mathcal{N}(0, 1)$, then

$$\frac{(X_1, X_2, ..., X_d)}{\sqrt{X_1^2 + X_2^2 + ... + X_d^2}}$$

is a direction chosen uniformly at random in a d-dimension space.

Also, many distributions can be approximated using normal distributions. For example, assume each $X_i, i \in [n]$ is chosen randomly from $\{-1, 1\}$ and consider $X = \sum_{i=1}^n X_i$. Notice that $\mathbb{E}[X] = 0$ and $\mathrm{Var}[X] = n$. Then, $\frac{X}{\sqrt{n}}$ is very close to the standard normal distribution $\mathcal{N}(0, 1)$.

3 Proof of JL-Theorem

We choose a matrix $M \in \mathbb{R}^{k \times d}$, where every entry $m_{i,j}$ is chosen independently from $\mathcal{N}(0, 1)$. Then we define the mapping f to be:

$$f(a) = \frac{Ma}{\sqrt{k}}.$$

We need to show that the f satisfies the property of the theorem statement with high probability. We fix a pair $i \neq j \in [n]$, and define $r = a_j - a_i$. Then $\|r\|$ denotes the distance between a_i and a_j. $\frac{Mr}{\sqrt{k}}$ is the distance between the two points in the projected space. We now compute the expected square length of Mr.

$$E[\|Mr\|^2] = \mathbb{E}\left[\sum_{i=1}^{k}\left(\sum_{j=1}^{d} m_{ij}r_j\right)^2\right] = \sum_{i=1}^{k}\mathbb{E}\left(\sum_{j=1}^{d} m_{ij}r_j\right)^2$$

$$= \sum_{i=1}^{k}\mathbb{E}\left(\sum_{j=1}^{d}\sum_{j'=1}^{d} m_{ij}m_{ij'}r_{ij}r_{ij'}\right) = \sum_{i=1}^{k}\sum_{j=1}^{d}\sum_{j'=1}^{d} r_j r_{j'}\,\mathbb{E}[m_{ij}m_{ij'}].$$

If $j \neq j'$, then we know $\mathbb{E}[m_{i,j}m_{i,j'}] = 0$ since $m_{i,j}$ and $m_{i,j'}$ are independent standard normal variables. If $j = j'$, then $\mathbb{E}[m_{i,j}m_{i,j'}] = \mathbb{E}[m_{i,j}^2] = \mathrm{Var}(\mathcal{N}(0, 1)) = 1$. Thus, we have

$$\mathbb{E}\left[\|Mr\|^2\right] = \sum_{i=1^k}\sum_{j=1}^{d} r_j^2 = k\|r\|^2.$$

Indeed, the above equality can be proved directly. Let M_i be the i-th row vector of M. Then $M_i r$ will be a normal variable with mean 0 and variance $\sum_{j=1}^d r_j^2 = \|r\|^2$. Thus, $\mathbb{E}[\|M_i r\|^2] = \|r\|^2$ for every $i \in k$ and $\mathbb{E}[\|Mr\|^2] = k\|r\|^2$.

3.1 Proof of Concentration Bound

In this section, we show

$$\Pr[\|Mr\|^2 > (1 + \varepsilon)k\|r\|^2] < e^{-\frac{\varepsilon^2 k}{12}}, \text{and} \quad (1)$$

$$\Pr[\|Mr\|^2 < (1 - \varepsilon)k\|r\|^2] < e^{-\frac{\varepsilon^2 k}{12}}. \quad (2)$$

using the analysis similar to the proof of Chernoff bounds. If we choose $k = O(\frac{\log n}{\varepsilon^2})$ to be large enough, then we have $e^{-\frac{\varepsilon^2 k}{12}} \leq \frac{1}{2n^2}$. Applying union bound, with probability at least $\frac{1}{2}$, for every i and j, we have

$$(1 - \epsilon)k\|a_j - a_i\|^2 \leq \|M(a_j - a_i)\|^2 \leq (1 + \epsilon)k\|a_j - a_i\|^2.$$

This will finish the proof of the JL theorem. Indeed, the proof gives a randomized algorithm to produce the matrix M.

We only prove Inequality (1); the proof of Inequality (2) will be symmetric. For every $i \in k$, let $X_i = M_i r$, where M_i is the i-th row vector of M. Notice that $\|Mr\|^2 = \sum_{i=1}^k X_i^2$. Let $t \in (0, 1/2)$ be any number; we have

$$\begin{aligned}
\Pr\left[\sum_{i=1}^k X_i^2 > (1+\epsilon)k\right] &= \Pr\left[e^{t\sum_{i=1}^k X_i^2} > e^{t(1+\epsilon)k}\right] \\
&< \frac{\mathbb{E}\left[e^{t\sum_{i=1}^k X_i^2}\right]}{e^{t(1+\epsilon)k}} && \text{by Markov Inequality} \\
&= e^{-t(1+\epsilon)k}\prod_{i=1}^k \mathbb{E}[e^{tX_i^2}] && \text{by independence of } X_i \text{ variables} \\
&= e^{-t(1+\epsilon)k}\left(\frac{1}{\sqrt{1-2t}}\right)^k && \text{this will be proved soon} \\
&= \left(\frac{1}{\sqrt{1-2t}\cdot e^{t(1+\epsilon)}}\right)^k.
\end{aligned}$$

We used that $\mathbb{E}[e^{tX_i^2}] = \frac{1}{\sqrt{1-2t}}$. This can be proved as follows:

$$\begin{aligned}
\mathbb{E}[e^{tX_i^2}] &= \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty} e^{-x^2/2}\cdot e^{tx^2}\mathrm{d}x = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty} e^{(t-\frac{1}{2})x^2}\mathrm{d}x \\
&= \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty} e^{-\frac{1}{2}y^2}\frac{\mathrm{d}y}{\sqrt{1-2t}} && \text{by letting } y = \sqrt{1-2t}\cdot x \\
&= \frac{1}{\sqrt{1-2t}}. && \text{since} \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty} e^{-\frac{1}{2}y^2}\mathrm{d}y = 1
\end{aligned}$$

We choose $t = \frac{\epsilon}{2(1+\epsilon)} \le \frac{1}{2}$. Then,

$$\begin{aligned}
\Pr\left[\sum_{i=1}^k X_i^2 > (1+\epsilon)k\right] &\le \exp\left(-\frac{k}{2}\left(2t(1+\epsilon) + \ln(1-2t)\right)\right) \\
&= \exp\left(-\frac{k}{2}\left(\epsilon - \ln(1+\epsilon)\right)\right) \\
&\le \exp\left(-\frac{k}{2}\left(\epsilon - \epsilon + \frac{\epsilon^2}{2} - \frac{\epsilon^3}{3}\right)\right) \\
&\le e^{-k(\epsilon^2/2-\epsilon^3/3)/2}.
\end{aligned}$$

Above, we used that $\ln(1+\epsilon) \le \epsilon - \frac{\epsilon^2}{2} + \frac{\epsilon^3}{3}$, which can be derived from the Taylor expansion of $\ln(1+\epsilon)$. If $\epsilon < 1$, then the probability is at most $e^{-k\epsilon^2/12}$. This finishes the proof of Inequality (1).

4 Application of JL Theorem in Streaming Algorithms

One application of the analysis of the JL theorem is designing a streaming algorithm for estimating the L_2-norm of a vector that comes online. Let U be huge universe. Every time an element $u \in U$ comes, we need to add it to the counter f_u, that indicates the frequency of u in the stream. We want to estimate $\|f\|_2^2 = \sum_{u\in U} f_u^2$.

Consider the following algorithm:

1: Randomly construct a matrix $M \in \mathbb{R}^{[k]\times U}$ for some k whose value will be decided later, where each entry is chosen independently and randomly from the normal distribution.

2: Output $\frac{\|Mf\|^2}{k}$ using the following procedure: Let $v \leftarrow (0,0,0,0,\ldots,0)^k$ initially. whenever u comes, we update $v \leftarrow v + M_u$, where M_u is the column of M correspondent to u. Then, we output $\|v\|^2/k$.

By the analysis of the JL theorem, we can show that if choose a large enough $k = O(\frac{\log(1/\delta)}{\epsilon^2})$, then the probability that our output is within $(1 \pm \epsilon)\|f\|^2$ is at least $1 - \delta$. However, the above algorithm does not work directly since we have to store the whole matrix M at the beginning: this is needed to make sure that we are using the same column for the same element u.

The issue can be handled using *pseudo-randomness*, which is a topic beyond the scope of this course. The idea is that we have a pseudo-random matrix generating algorithm $\mathcal{A}$, which takes a small length string s and output a matrix M. Then, in our algorithm, we randomly choose the string s and store it in memory. Whenever we need to use M_u, we simply call the algorithm $\mathcal{A}(s)$ and use the u-th columns of the output matrix (to save time, we do not need $\mathcal{A}$ to output all columns, just the u-th column). In other words, we pretend that the distribution of the matrix that $\mathcal{A}$ outputs is similar to the one generated using normal distributions. Psudo-randomness theory says that there exists such an algorithm $\mathcal{A}$ such that an time or space restricted algorithm (including our algorithm for estimating $\|f\|^2$) will not be able to distinguish between the two distributions; in other words, the pseudo-random matrix generator $\mathcal{A}$ is able to "fool" our algorithm. Thus, the streaming algorithm will have almost the same behavior as if the matrix was generated using normal distributions.