Colorado CSCI 5454: Algorithms November 7, 2019

Lecture 22

Lecturer: Bo Waggoner *Scribe: Bo Waggoner*

The Johnson-Lindenstrauss Lemma

1 Dimensionality Reduction

We have a collection of vectors $x_1, \ldots, x_n$ in some very high-dimensional space, i.e. $\mathbb{R}^m$ for very large m. Many common algorithmic tasks involving this data set scale poorly with n. For example, if we collect these into an $n \times m$ matrix, then matrix multiplication or inversion, or things like computing the singular value decomposition (SVD), can be very computationally expensive.

The goal is to summarize this data in a much lower-dimensional space. In particular, we will project each x_i down to a point $y_i \in reals^d$ for some relatively small d.

However, we want the structure of the data to be preserved. In particular, this lecture will focus on the *pairwise distances*. That is, we want $\|y_i - y_j\| \approx \|x_i - x_j\|$ for all pairs i, j.

1.1 The Statement of the Lemma

The Gaussian projection. Given integers m and d, let $A \in \mathbb{R}^{d \times m}$ be a random matrix where each entry $A(i,j)$ is distributed independently Normal$(0,1)$ (recall this is a Gaussian variable with mean 0 and variance 1). Let $B = \frac{1}{\sqrt{d}} A$, i.e. each entry of A is scaled by $\frac{1}{\sqrt{d}}$. Let $y_i = Bx_i$ for each $i = 1, \ldots, n$.

Lemma 1 (Example JL Lemma)**.** *Let $x_1, \ldots, x_n \in \mathbb{R}^m$ be any collection of points. Let $d \geq \frac{8}{\epsilon^2} \ln\left(\frac{n}{\delta}\right)$ and let $y_1, \ldots, y_n$ be the results of the Gaussian projection defined above. Then for any $\epsilon \in (0, 1]$, with probability at least $1 - \delta$, we have for all i, j that*

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

The proof will be broken up into pieces next. Note that if $x_i = x_j$, then $y_i = y_j$ and the pairwise distance is exactly preserved, so we focus on the case where all x_i are distinct.

1.2 Proving that norms are preserved

Lemma 2. *Let $z \in \mathbb{R}^m$ be any unit vector and $B \in \mathbb{R}^{d \times m}$ the random Gaussian projection defined above. For any $\epsilon \in (0, 1]$, we have $(1 - \epsilon) \leq \|Bz\| \leq (1 + \epsilon)$ except with probability at most $2e^{-d\epsilon^2/4}$.*

Proof. We have

$$\begin{aligned}
\|Bz\|^2 &= \sum_{i=1}^{d} ((Bz)(i))^2 \\
&= \frac{1}{d} \sum_{i=1}^{d} ((Az)(i))^2 \\
&= \frac{1}{d} \sum_{i=1}^{d} \left(\sum_{j=1}^{m} A(i,j) z(j) \right)^2 .
\end{aligned}$$

Let $w(i) := \sum_{j=1}^{m} A(i,j)z(j)$. By the properties of the normal distribution (Fact 1), $w(i)$ is distributed $\text{Normal}\left(0, \sum_{j=1}^{m} z(j)^2\right) = \text{Normal}(0,1)$ because z is a unit vector. And

$$\|Bz\|^2 = \frac{1}{d}\sum_{i=1}^{d} w(i)^2.$$

Now, $\sum_{i=1}^{d} w(i)^2$ is distributed chi-squared(d). Therefore, by a tail bound (Fact 2), we have

$$\Pr\left[\sum_{i=1}^{d} w(i)^2 \geq (1+\epsilon)^2 d\right] \leq e^{-d\epsilon^2/4}$$
$$\Pr\left[\sum_{i=1}^{d} w(i)^2 \leq (1-\epsilon)^2 d\right] \leq e^{-d\epsilon^2/4}.$$

So except with probability $2e^{-d\epsilon^2/4}$, we have $\|Bz\|^2 \leq (1+\epsilon)^2$ and $\|Bz\|^2 \geq (1-\epsilon)^2$, which proves the claim. □

We used the following facts:

Fact 1. *Let $X \sim$Normal$(0,\sigma_1^2)$ and $Y \sim$Normal$(0,\sigma_2^2)$, independently. Then αX is distributed Normal$(0,\alpha^2\sigma_1^2)$; and $X+Y$ is distributed Normal$(0,\sigma_1^2+\sigma_2^2)$.*

Fact 2. *Suppose W is distributed chi-squared(d), i.e. the sum of d independent Normal$(0,1)$ variables. Then for any $\epsilon < 1$, we have:*

$$\Pr[Z \geq (1+\epsilon)^2 d] \leq e^{-d\epsilon^2/4}$$
$$\Pr[Z \leq (1-\epsilon)^2 d] \leq e^{-d\epsilon^2/4}.$$

Proof. We cite a more common form of the tail bound, e.g. Laurent and Massart 2000: for any $t \geq 0$, we have

$$\Pr[Z \geq d + 2\sqrt{dt} + 2t] \leq e^{-t}$$
$$\Pr[Z \leq d - 2\sqrt{dt}] \leq e^{-t}.$$

Given this, let $t = d\epsilon^2/4$. In other words, $\epsilon = 2\sqrt{t/d}$. Then

$$\begin{aligned}(1+\epsilon)^2 d &= d + 2d\epsilon + d\epsilon^2 \\ &= d + 4\sqrt{dt} + 4t \\ &\geq d + 2\sqrt{dt} + 2t.\end{aligned}$$

This gives

$$\begin{aligned}\Pr[Z \geq (1+\epsilon)^2 d] &\leq \Pr[Z \geq d + 2\sqrt{dt} + 2t] \\ &\leq e^{-t} \\ &= e^{-d\epsilon^2/4}.\end{aligned}$$

Similarly:

$$\begin{aligned}(1-\epsilon)^2 d &\leq (1-\epsilon)d \\ &= d - \epsilon d \\ &= d - 2\sqrt{dt}.\end{aligned}$$

This gives $\Pr[Z \leq (1-\epsilon)^2 d] \leq \Pr[Z \leq d - 2\sqrt{dt}] \leq e^{-t} = e^{-d\epsilon^2/4}$. □

1.3 Union-bounding

Proof of Lemma 1. Set $d \geq \frac{8}{\epsilon^2} \log \frac{n}{\delta}$. Consider any pair of points i, j. Let $z_{ij} = \frac{x_i - x_j}{\|x_i - x_j\|}$. Note z_{ij} has norm 1.

The key point is that

$$\begin{aligned} Bz_{ij} &= \frac{Bx_i - Bx_j}{\|x_i - x_j\|} \\ &= \frac{y_i - y_j}{\|x_i - x_j\|}. \end{aligned}$$

By Lemma 2, we have except with probability $2e^{-d\epsilon^2/4}$,

$$\begin{aligned} & 1 - \epsilon \leq \|Bz_{ij}\| \leq 1 + \epsilon \\ \iff \quad & (1-\epsilon)\|x_i - x_j\| \leq \|y_i - y_j\| \leq (1+\epsilon)\|x_i - x_j\|. \end{aligned}$$

By a union bound[1], this holds for all $\binom{n}{2}$ pairs of points except with probability

$$\begin{aligned} \frac{n(n-1)}{2}\left(2e^{-d\epsilon^2/4}\right) &\leq n^2 \exp\left(\frac{-d\epsilon^2}{4}\right) \\ &\leq n^2 \exp\left(\frac{-\left(\frac{8\ln(n/\delta)}{\epsilon^2}\right)\epsilon^2}{4}\right) \\ &= n^2 \exp\left(-\ln(n^2/\delta^2)\right) \\ &= n^2 \frac{\delta^2}{n^2} \\ &= \delta^2 \\ &\leq \delta. \end{aligned}$$

□

2 Discussion and Applications

There are several remarkable aspects of this lemma.

1. We simply projected the points down in a linear and random way, not taking into account the structure of the points at all. Yet the dimensionality turns out to be optimal, if one still requires preserving pairwise distances. (In fact, many other linear, random projections, besides indpenedent Gaussians, give similar guarantees.)

2. The final dimensionality needed, d, only depended on the *number of points* n and not on the original dimension of the space they resided in!

3. Furthermore, the dimensionality only depends logarithmically on the number of points.

One example where dimension is high is images. An image of 1000 by 1000 pixels with 3 color values per pixel lives in a 3-million-dimensional space. Computing even inner products or measuring distance between points in this space is a significant computational overhead.

[1] Recall this says that the probability any of the events happened is at most the sum of their probabilities.

Application: nearest neighbors. In this task, we have a database of n points $x_1, \ldots, x_n$. We are then repeatedly given a new point, x^*, and asked to find the nearest x_i to x^*. However, if points live in a high-dimensional space, the computation time can be quite expensive.

If we first pick and store a JL matrix B, along with all the projected points $y_1, \ldots, y_n$, then we can answer new queries quickly: Compute $y^* = Bx^*$, and find the closest y_i to y^*. This will be correct up to a factor of $(1 \pm \epsilon)$ compared to the actual closest point, and will be much faster to compute if the dimensionality is much lower.

Nearest neighbors can be useful for any task where we want to find similar objects. For example, suppose we have a data set of people's preferences for e.g. restaurants or movies. Each person's preferences is a point x_i in high dimensional space, where $x_i(j)$ represents a rating of item j. Given a new person, we can ask to find an existing person who's most similar (or a group).

Application: clustering. In this task, we have a similar database, but the goal is to partition it into k *clusters*, where each cluster is a group of points that are all relatively close to each other. By projecting the points into a lower dimension first, we can approximately preserve all pairwise distances, yet compute the clustering much more quickly.

Clustering is useful extremely broadly for making sense of large data sets. Going back to the recommendations example, we might use a clustering problem to group our dataset of people's preferences into k clusters to find groups of similar people. The centroid (average of the points) of the cluster is some representation of a "typical" person in the cluster.

A popular starting algorithm for clustering is k-means:

1. Pick an initial set of k means $\mu_1, \ldots, \mu_k$, e.g. by subselecting k of the points randomly.
2. Pick a point x_i.
3. Let $\mu_j = \arg\min_{j'} \|x_i - \mu_{j'}\|$, i.e. the closest mean to x_i.
4. Assign x_i to cluster j (removing it from its current cluster if necessary).
5. Recompute the means $\mu_1, \ldots, \mu_k$.
6. Pick a new point x_i and repeat, e.g. by iterating through the points in some order.