Lecture 5. Dimensionality Reduction. Linear Algebra

Ricard Gavaldà

MIRI Seminar on Data Streams, Spring 2015

Contents

1. Dimensionality reduction
   - Matrix product
   - Metric space embeddings
   - Linear regression
   - *k*-means clustering
2. Matrix sketches
   - SVD
   - Frequent Directions

Dimensionality reduction

Dimensionality reduction

Out there, there is a large matrix $M \in \mathbb{R}^{n \times m}$

Dimensionality reduction

Can we instead keep a smaller $M' \in \mathbb{R}^{n' \times m'}$ with $n' \ll n$ or $m' \ll m$ or both, so that computing on M' gives results similar to computing on M?

Applications:

- Information Retrieval - bag of words models for documents
- Machine learning - reducing instances or attributes
- PCA - Principal Component Analysis
- Clustering with many objects or many dimensions
- Image Analysis

Matrix product

Approximate matrix product, nonstreaming

- Matrices $A \in \mathbb{R}^{n \times p}$ and $B \in \mathbb{R}^{p \times m}$, want AB
- Build "random matrix" $S \in \{+1, -1\}^{k \times p}$
  (we called this "k hash functions" before)
- Approximate AB by $(AS^T) \cdot (SB)$
- I.e., $(AB)[i,j] \simeq \sum_\ell (AS^T)[i,\ell](SB)[\ell,j]$

Saves computation if $k \ll p, n, m$ because

$$npk + kpm + nkm \ll npm$$

Matrix product

Claim

If $k = O(\varepsilon^{-1}\ln(n/\delta))$, with probability $1-\delta$

$$\|AB-(AS^T)(SB)\|_F \leq \varepsilon\|A\|_F\|B\|_F$$

where $\|A\|_F = \sqrt{\sum_{i,j} A_{i,j}^2}$ is the Frobenius norm

We have already seen the proof before (essentially)
Here absolute, instead of relative, error $k \simeq 1/\varepsilon$

Matrix product

Goal:

$$\|AB - (AS^T)(SB)\|_F \leq \varepsilon \|A\|_F \|B\|_F$$

1. $E[\langle Sx, Sy \rangle] = \langle x, y \rangle$, for every x, y
2. Then $E[(AS^T)(SB)] = AB$
3. $Var[\langle Sx, Sy \rangle] \leq 2\varepsilon^2 \|x\|_F^2 \|y\|_F^2$

   (averaging of k rows & Chebyshev hidden here!)
4. Then $Var[\,\|AB - (AS^T)(SB)\|_F\,] \leq 2\varepsilon^2 \|A\|_F^2 \|B\|_F^2$
5. Median trick: Run $O(\ln(1/\delta))$ copies of the above
   - Complication: “median of matrices” is undefined
   - Idea: Find an estimate that is close to most others
   - Estimate $d = \|A\|_F^2 \|B\|_F^2$ for each, from sketches
   - Return an estimate closer than $d/2$ to more than half the rest

Matrix product

Approximate AB by $(AS^T) \cdot (SB)$, streaming:

- build sketches for every row of A and every column of B
- Easy to update sketch AS^T when new entry of A arrives
- Easy to update sketch SB when new entry of B arrives

Metric space embeddings

- We are mapping a large space to a smaller space
- Variance is reduced using min or median-of-averages
- This is not a metric: does not preserve usual distances
- More general ways of saying

  "We embed our dimension k space into a dimension k' space, $k' < k$ that preserves *metrics*"
- More problem-independent, geometric (and interesting)

Metric space embeddings

Reminder: a metrid d satisfies $d(x,y) = d(y,x) \geq 0$, with $d(x,x) = 0$ only, and triangle inequality $d(x,y) + d(y,z) \geq d(x,z)$

An embedding f from metric space (X, d_X) to metric space (Y, d_Y) has *distortion* ε if for every $a, b \in X$

$$(1-\varepsilon)d_X(a,b) \leq d_Y(f(a), f(b)) \leq (1+\varepsilon)d_X(a,b)$$

Metric space embeddings

[Johnson-Lindenstrauss 84]

Every n-point metric space can be embedded into ℓ_2^k with ε distortion, for $k = O(\varepsilon^{-2} \log n)$

- In words, the embedding is a map $X \to \mathbb{R}^k$
- Independent of dimension of original space!
- Basically only possible for L_2, not other L_p. We may still be able to approximate:

  $$\text{for all } x, y \in S,\ \|x - y\|_p \in (1 \pm \varepsilon) d(x, y)$$

  but then d is not a metric

Metric space embeddings

[Johnson-Lindenstrauss 84]

Every n-point metric space can be embedded into ℓ_2^k with ε distortion, for $k = O(\varepsilon^{-2} \log n)$

- Not just existential result: holds for *most* mappings defined by 1) independent $\{+1, -1\}$ entries, 2) independent $N(0, 1)$ entries
- But such matrices are not sparse: updates are computationally costly
- Many deep papers on computationally lighter variants (Fast Johnson-Lindenstrauss, enforcing sparsity, ...)

Linear regression (least squares)

Given n pairs $(x_i, y_i) \in \mathbb{R}^{d+1}$, find $r \in R^d$ that minimizes

$$\sum_{i=1}^{n} (y_i - r \cdot x_i)^2$$

Alternatively,

Given $A \in \mathbb{R}^{n \times d}$ and $b \in R^n$, find x that minimizes $\|Ax - b\|_2$

Method:

- Minimize in sketch space
- Memory $O(d^2/\varepsilon^2 \ln(n/\delta))$

k-means clustering

Given $x_1, \ldots, x_n \in \mathbb{R}^d$,

$$\operatorname*{argmin}_{C_1,\ldots,C_k} \sum_{i=1}^{n} \|x_i - C_{x_i}\|_2$$

Use random $S \in \mathbb{R}^{d \times r}$

- Minimize in sketch space
- Can be shown to preserve value of optimal solution to factor $1 \pm \varepsilon$ for $r = O(k/\varepsilon^2 \log(n/\delta))$

Matrix sketches

Singular Value Decomposition

At the heart of many techniques:

- Principal Component Analysis
- Spectral Clustering
- Data Compression
- Latent Semantic Indexing
- Latent Dirichlet Allocation
- Spectral methods for HMM
- ...

Singular Value Decomposition

For $A = UDV^T \in \mathbb{R}^{n \times n}$

$$A = \left(\begin{array}{c|c|c} & & \\ u_1 & \cdots & u_n \\ & & \end{array} \right) \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \end{pmatrix} \left(\begin{array}{c} v_1 \\ \hline \vdots \\ \hline v_n \end{array} \right)$$

Intuition: If A is a document - term incidence matrix:

- (at most n) hidden topics
- U tells how close each document is to each topic
- V tells how close each term is to each topic
- D measures the presence of each topic

Singular Value Decomposition

SVD theorem

Let $A \in \mathbb{R}^{n \times m}$. There are matrices $U \in \mathbb{R}^{n \times n}$, $D \in \mathbb{R}^{n \times m}$ and $V \in \mathbb{R}^{m \times m}$ such that:

- $A = UDV^T$
- U and V are orthonormal: $U^T U = I \in \mathbb{R}^{n \times n}$ and $V^T V = I \in \mathbb{R}^{m \times m}$
- D is a diagonal matrix of non-negative real numbers
- Additionally, $A = \sum_i \sigma_i u_i v_i^T$

- The diagonal values of D, denoted $\sigma_1, \sigma_2, \ldots$, are the *singular values* of A; w.l.o.g. $\sigma_1 \geq \sigma_2 \geq \ldots$
- It follows that $\text{rank}(A) = \text{rank}(D)$ is the number of non-zero singular values
- Column vectors of U (V) are its *left (right) singular vectors*

SVD and low-rank approximation

Choose $k \leq n, m$

Let D_k be the result of keeping retaining the first k diagonal values in D and zeroing the rest,

That leaves only the heaviest k "components"

Fact

$A_k = UD_kV$ is the best rank-k approximation of A.

I.e., A_k minimizes $\|A - B\|_F$ among all rank-k matrices B

Singular Value Decomposition

The SVD decomposition can be computed in time $O(nm^2)$
But the power method is often preferred:

- Define $M = A^T A$
- Take repeated powers of M
- If $\sigma_1 > \sigma_2$, M^t approaches $\sigma_1^{2t} v_1^T v_1$
- which leads to σ_1 and v_1
- Subtract, repeat, to get other values

So a sketch for $A^T A$ is good for sketching $SVD(A)$, which is good for sketching A

Matrix sketches

Given ε and a matrix $A \in \mathbb{R}^{m \times n}$, want to keep a sketch $B \in \mathbb{R}^{k \times n}$ such that e.g.

$$\|B^T B - A^T A\|_F \leq \varepsilon \|A\|_F^2$$

Approaches:

- Dimensionality reduction - hashing. Space $O(n/\varepsilon^2)$
- Column or row sampling. Space $O(n/\varepsilon^2)$
- Frequent directions [Liberty13]. Space $O(n/\varepsilon)$

Remember: Random Sampling for frequent elements?

Take a random sample from the stream, estimate item frequency in sample, compute hotlist

- Problem 1. Bad for top-k. Misses many small elements
- Problem 2. Anyway, how to keep a uniform sample?
- (Solution to 2.) Reservoir sampling [Vitter85]
  - ...
- Even for heavy hitters, required sample size is $O(1/\varepsilon^2)$
- But $O(1/\varepsilon)$ solutions exist

Matrix sketches by sampling

- Fix k, number of rows (or columns) to keep
- Decide each row (or column) with probability proportional to its L_2 norm
- If $k = O(1/\varepsilon^2)$, this gives a matrix B such that

$$\|B^T B - A^T A\|_F \leq \varepsilon \|A\|_F^2$$

- Quite nontrivial to get tight bounds

Simple deterministic matrix sketching - Frequent Directions

[Liberty13]

- Inspired by the heavy hitter algorithms - [KPS] in particular
- Gets memory bound $O(n/\varepsilon)$ instead of $O(n/\varepsilon^2)$
- Is deterministic
- Performs better (accuracy-wise) than hashing and sampling for given memory; slightly slower updates

Idea:

Instead of storing “frequent items” we store “frequent directions”

A variant of [KPS]

Table $(K, count)$; it's never full

Update(x):

if $x \in K$ then $count[x]++$

else

add x to K with count 1;

if $|K| = k$ then

remove the $k/2$ elements with lowest counts;

Intuition: each symbol occurrence discounts k occurrences. Therefore, at most t/k occurrences of any a not counted in count

A variant of [KPS]

Table (K, count); it's never full

Update(x):

if $x \in K$ then $\mathit{count}[x]{+}{+}$

else

add x to K with count 1;

if $|K| = k$ then

remove the $k/2$ elements with lowest counts;

Fact: At any time t, for every x, not even in K,

$$\mathit{freq}_t(x) - \mathit{count}[x] \leq 2t/k$$

The spirit of Frequent Directions

Matrix B, initially all 0

Update(A_i): // A_i is ith row of A

insert A_i into zero-valued row of B;

if (B has no zero-valued rows)

rotate rows of B so that they are orthogonal;

remove the $k/2$ lightest rows

Intuition [Liberty13]: 'The algorithm "shrinks" k orthogonal vectors by roughly the same amount. This means that during shrinking steps, the squared Frobenius norm of the sketch reduces k times faster than its squared projection on any single direction'

Frequent Directions

Matrix B, initially all 0

Update(A_i): // A_i is ith row of A

insert A_i into zero-valued row of B;

if (B has no zero-valued rows)

$[U, D, V] \leftarrow SVD(B)$;

$\sigma \leftarrow \sigma_k^2$;

$D' \leftarrow \sqrt{\max(D^2 - I_k \sigma, 0)}$;

$B \leftarrow D' V^T$; // at least half the rows of B are set to 0

Fact: At any time t,

$$\|B^T B - A^T A\|_F \leq 2\|A\|_F^2 / k$$

Frequent Directions

Running time

- dominated by SVD(B) computation, $O(nk^2)$
- but this is every $k/2$ rounds
- $\therefore$ amortized $O(nk)$ per row
- (reasonable: n is row size)

Observation: Easy to parallelize

- Sketch separately disjoint sets of rows
- Then stack sketches and sketch that matrix