CS 596: Theoretical Machine Learning

Lecturer: Pranjal Awasthi
Scribe: Kaitlin Poskaitis

Lecture #13
November 22, 2016

1 Randomized SVD

So far, we have used SVD to solve several problems such as stochastic block models and matrix completion. Also note that in all such problems we typically only need to compute k-SVD, i.e., the top k components in the decomposition. Furthermore, we are happy with an approximate k-SVD rather than an exact one. In this lecture we will see fast randomized algorithm for computing such approximate k-SVD decompositions.

Problem Statement

Input:
$A_{n\times n}, k$
Output:
A' s.t. $rank(A') = k$,
$||A - A'||_F \leq \min_{B:rank(B)=k} ||A - B||_F + error$

One solution to this problem is to compute the full SVD. If the spectrum of A decays nicely, the power method can be used. It it does not, then QR decomposition needs to be used, which has a runtime of $O(n^3)$. This is often prohibitively expensive. When developing a more efficient algorithm, there are a few parameters that we should keep in mind. First and foremost, the error should be small. The runtime should be (near) linear with respect to $nnz(A)$, where $nnz(A)$ is the number of non-zero entries in A. Our approach should be parallelizable and it should only take 1-2 passes over A to complete (in case A doesn't fit in memory).

Approach

Our approach to this problem will be as follows. First, take A and compress it to get a smaller matrix $B_{n\times l}$ where $l << n$. Next, compute the full SVD of B to get $B' = \sum_{i=1}^{n} \lambda_i a_i b_i^T$. Now use this decomposition to output a good approximation to k-SVD of A. Let's first see what should we output. Let $A = \sum_{i=1}^{n} \sigma_i u_i v_i^T$. Then the ideal approximation is $A' = \sum_{i=1}^{k} \sigma_i u_i v_i^T = (\sum_{i=1}^{k} u_i u_i^T)(\sum_{i=1}^{n} \sigma_i u_i v_i^T) = PP^T A$, where $P = [u_1, u_2, \ldots, u_k]$. Instead we will output $= (\sum_{i=1}^{k} a_i a_i^T)A = QQ^T A$ where $Q = [a_1, a_2, \ldots, a_k]$. As long as the subspace spanned by the first k singular vectors of B is correlated with that of the first k singular vectors of A, B' will approximate A'.

Compression Scheme

Let $R_{n\times l}$ be a matrix such that $R_{ij} = N(0, 1)$. Let $B' = AR$. The intuition behind this is as follows. Assume A is rank-k. In order to get k linearly independent vectors from A, take k random linear combinations by multiplying A by a random matrix. Since A is not necessarily rank-k, there might be some noise in this output.

Recall our proposed algorithm from earlier:

Step 1: Compression: $B = AR$. In this step, we take l matrix-vector products. This is highly parallelizable and only requires one pass over A.

Step 2: SVD of an $n \times l$ matrix. $O(nl^2)$ time.

Step 3: $(\sum_{i=1}^{k} a_i a_i^T)A$. $O(nl^2)$ time.

So the algoritm is very efficient if we can keep l small and achieve low error.

Theorem 1 (Johnson-Lidenstraus Lemma)**.** *Let $v \in R^n$ be a fixed vector and , $R_{n\times l}$ is a random $N(0,1)$ matrix. Then, $E[||v^T R||^2] = l||v||^2$ and*

$P(||v^T R||^2 > (1+\epsilon)l||v||^2) \leq 2\sqrt{l}e^{\frac{-l\epsilon^2}{4}}$

$P(||v^T R||^2 > (1-\epsilon)l||v||^2) \leq 2\sqrt{l}e^{\frac{-l\epsilon^2}{4}}$

Choosing $l = \frac{10logn}{\epsilon^2}$ and using the above theorem with a union bound we get that w.p. at least $\frac{9}{10}$ the following holds

- For each row A_i of A, $\|A_i R\|^2 \in (1 \pm \epsilon)l\|A_i\|^2$.
- For each column A^i of A, $\|(A^i)^T R\|^2 \in (1 \pm \epsilon)l\|A^i\|^2$.
- For each left singular vector u_i of A, $\|u_i^T R\|^2 \in (1 \pm \epsilon)l\|u_i\|^2$.
- For each right singular vector v_j of A, $\|v_j^T R\|^2 \in (1 \pm \epsilon)l\|v_j\|^2$.

The algorithm is the following

- Let $B' = \frac{1}{\sqrt{l}}AR = \sum_{i=1}^{n} \lambda_i a_i b_i^T$.
- Output $A' = (\sum_{i=1}^{k} a_i a_i^T)A$.

Theorem 2. *With probability $\geq \frac{9}{10}$ over the choice of R, the following holds: $||A - A'||_F^2 \leq (1+\epsilon)\min_{B:rank(B)=k} ||A - B||_F^2$*

We will prove a weaker statement where $A' = (\sum_{i=1}^{k} a_i a_i^T)A$ and the guarantee will be $||A - A'||_F^2 \leq \min_{B:rank(B)=k} ||A - B||_F^2 + \epsilon||A||_F^2$

Proof. The proof will rely on Lemma 1 stated at the end that shows that most of the signal in the first k components of A is present in the first $2k$ components of B'.

We have $||A - A'||_F^2 = \sum_{i=1}^{n} ||a_i^T(A - A')||^2$

$= \sum_{i=1}^{2k} ||a_i^T(A - A'||^2 + \sum_{i=2k+1}^{n} ||a_i^T(A - A')||^2$

For $i \leq 2k$:

$a_i^T(A - A') = a_i^T A - a_i^T(\sum_{j=1}^{2k} a_i a_j^T)A = 0$

For $i > 2k$:

$a_i^T(A - A') = \sum_{i=2k+1}^{n} ||a_i^T A||^2$

$= \sum_{i=1}^{n} ||a_i^T A||^2 - \sum_{i=1}^{2k} ||a_i^T A||^2$

$= ||A||_F^2 - \sum_{i=1}^{2k} ||a_i^T A||^2$

Hence we get that $\|A - A'\|^2 = ||A||_F^2 - \sum_{i=1}^{2k} ||a_i^T A||^2$. Also $||A - B||_F^2 = \sum_{i=k+1}^{n} \sigma_i^2 = ||A||_F^2 - \sum_{i=1}^{k} \sigma_i^2$. Hence the error $\Delta = \sum_{i=1}^{k} \sigma_i^2 - \sum_{i=1}^{2k} ||a_i^T A||^2$.

(1): $\lambda_i^2 = ||a_i^T B'||^2$

$= ||a_i^T(\frac{1}{\sqrt{l}})AR||^2$
$= ||(\frac{1}{\sqrt{l}})(a_i^T A)R||^2$
$\leq (1+\epsilon)||a_i^T A||^2$ [follows from the Johnson-Lindenstrauss theorem]

(1) implies:
$\Delta \leq \sum_{i=1}^{k} \sigma_i^2 - \sum_{i=1}^{2k} \frac{\lambda_i^2}{1+\epsilon}$
$\Delta \leq \sum_{i=1}^{k} \sigma_i^2 - \frac{1-\epsilon}{1+\epsilon} \sum_{i=1}^{k} \sigma_i^2$
$\Delta \leq \frac{2\epsilon}{1+\epsilon} \sum_{i=1}^{k} \sigma_i^2$
$\Delta \leq 2\epsilon||A||_F^2$

□

Lemma 1. $\sum_{i=1}^{2k} \lambda_i^2 \geq (1-\epsilon) \sum_{i=1}^{k} \sigma_i^2$

Proof. See reference 1 below. □

Extensions

The prototype algorithm that we are using here is as follows. First, take A and multiply it by R to get B. Compute the full SVD of B and use this result to approximate A'. The fastest algorithms work by picking R to be specialized sparse matrices that have only has 1 non-zero entry per column resulting in runtime of $nnz(A) + O(\frac{nk^2}{\epsilon^3})$.

2 Additional Readings

- One of the earliest paper on randomized SVD. `https://www.math.cmu.edu/~af1p/Texfiles/SVD.pdf`.
- Better analysis for JL matrices. `http://dl.acm.org/citation.cfm?id=1170496`.
- A nice overview of randomized methods for SVD. `http://users.cms.caltech.edu/~jtropp/papers/HMT11-Finding-Structure-SIREV.pdf`.
- A recent paper providing the fastest algorithm for k-SVD. `https://arxiv.org/abs/1207.6365`.