ALG-ML SEMINAR

LOW-RANK WEIGHTED SVD MARCH 1, 2016

LECTURER: YUANZHI LI SCRIBE: KIRAN VODRAHALLI

Contents

1 Introduction

Abstract: We consider the problem of recovering low-rank matrices from noisy observations, when the noise is not uniformly distributed. This problem is naturally formulated as weighted low rank approximation: finding a rank-k matrix M' that minimize $\|MM'\|_{W^2} = \sum_{i,j} W_{i,j}(M_{i,j}M'_{i,j})^2$ for a given matrix M. Weighted low rank is a generalization of many standard problems such as k-SVD and matrix completion: when the $W_{i,j}$ are all one, the problem reduces to k-SVD. When $W_{i,j}$ are zero-one, the problem reduces to matrix completion. In this talk, I will survey the previous works on matrix completion and k-SVD, and present our new result of solving weighted low rank approximation with a vanilla alternative minimization algorithm. Under moderate assumptions (natural generalizations of the standard ones of matrix completion), our algorithm provably converges to ground truth M with a geometric rate from random initial point.

Today I will talk about a lot of results. Basically the talk is very high-level and mostly about background.

1.1 Weighted Low-Rank Problem Statement

The problem is the very simple problem of weighted low-rank approximation. This means you're given a matrix M and a weight matrix $W \in \mathbb{R}^{n\times n}$. You can consider $m \times n$ as well. For simplicity we will focux on $n \times n$ case. We want to find $\tilde{M}$ of rank k such that $\|M - \tilde{M}\|_W$ is small. So what is this norm? It means $\sum_{i,j} W_{ij}(M_{ij} - \tilde{M}_{ij})^2$. You're given matrices with some weights, and you want to find $\tilde{M}$ to minimize this weighted norm.

Now let's list some specific instances of this problem.

1. Now suppose $W_{ij} = 1$, $k = 1$, where we do things with respect to Frobenius norm:

$$\|M - xx^T\|_F$$

Well, this is just the largest eigenvector.

2. Suppose we again have $W_{ij} = 1$, $k > 1$, and we again minimize $\|M - \tilde{M}\|_F$ such that $\text{rank}(\tilde{M}) > k$. Well this is just k-SVD.

A lot of the interesting cases are when $W_{ij} \neq 1$ for all i, j.

1. $$W_{ij} = \begin{cases} 1 \\ 0 \end{cases}$$

   This means that we only care about some of the entries in the matrix. It basically means $\tilde{M}$ only need to match M at the points (i, j) where $W_{ij} = 1$. This problem is just **matrix completion**.

2. Now we are motivated by word vector embeddings: choose W_{ij} arbitrary. Let's say you take a word and map it to a vector in dimension $\mathbb{R}^{300}$. This vector has the following property:

   $$\langle w_1, w_2 \rangle \approx \log \frac{\mathbf{P}\{\text{word } 1, \text{word } 2\}}{\mathbf{P}\{\text{word } 1\} \cdot \mathbf{P}\{\text{word } 2\}}$$

   So how does this relate to weighted low-rank? Well, we don't really know what the probability of the words 1 and 2. We can look at a document to get estimates of what the true probabilities are. If you do it very carefully, you can see the optimal estimator is to minimize

   $$\sum_{i,j} f_{ij} \left(\langle w_i, w_j \rangle - \log \frac{P_{ij}}{P_i P_j} \right)^2$$

   where f_{ij} is associated with the noise. For P_{ij} large, the estimation is accurate. If P_{ij} is small (maybe rare words), the estimation is inaccurate. High accuracy means $(1 \pm \epsilon)P_{ij}$, when you take logs things become additive. So additive noise is very small. If you consider maximum likelihood estimation, you should reweight this sum so that $f_{ij} \sim P_{ij}$, since if P_{ij} is large, your noise is small and thus you're more accurate. So you can see this is exactly weighted low-rank matrix estimation. And moreover, we're not talking any distributional assumptions. It's very hard to say where the weight is coming from.

So this last case is the motivation, and remember the word vector problem as we continue the talk.

2 Known Past Hardness Results

Surprisingly, this problem is extremely hard even for rank 1 case. When $W_{ij} \geq 0$ and $k = 1$, this is NP-hard. You can very easily encode max-cut and other things into this problem. For matrix completion, it is not known to be NP-hard (though it is hard if certain other things are true, like finding cycles in k-colorable graphs or something like that)).

2.1 Largest Eigenvector Complexity

Finding the largest eigenvector is a subcase of weighted low-rank. The first thing you might try for finding the largest eigenvector in the weighted setting is power method. The running time you'll get is in terms of the number of nonzero vectors: $\mathcal{O}(nnz(M) \log(n/\epsilon) \frac{\lambda_1}{\lambda_2})$, where λ_1, λ_2 are the top two eigenvalues. You can do an accelerated version of the Power method, the Lanczos power method: $\mathcal{O}(nnz(M) \log(n/\epsilon) \sqrt{\frac{\lambda_1}{\lambda_2}})$. Then there is an interesting result by Clarkson and Woodruff giving a bound with the stable rank: $\mathcal{O}(nnz(M) + n \frac{\text{srank}(M)}{\max(\epsilon/k^2, \epsilon^{2.5})})$, which is not linear in ϵ. This has a lot of citations though. Then finally there is a method using SVRG, which has complexity $\mathcal{O}\left(\left(nnz(M) + n\text{srank}(M)^2k^2\right)\log(1/\epsilon)\log\log(1/\epsilon)\right)$. This is the original SVRG.

Finally, the latest result is by Tin, Kakade, Musro, Sidford. They improve the result from the stable rank squared to just the stable rank:

$$\mathcal{O}\left(\left[nnz(M) + n\text{srank}(M)k^2\right]\left[\log(1/\epsilon) + \log(kn)\right]\right)$$

There is also a slight improvement here, but let's forget it now. Then finally there is a paper by Garber and Hazan (Online Learning Eigenvectors). Hazan: We actually merged the paper, which is actually the same result: $\mathcal{O}\left(nnz(M)^{3/4}n^{1/4}\frac{1}{\sqrt{\epsilon}}\right)$.

2.2 k-SVD

Let's move on to k-SVD known results. There are only two results of which I am aware. Clarkson and Woodruff get

$$\|M - \tilde{M}\|_F \leq (1 \pm \epsilon)\text{opt}$$

where $\tilde{M}$ is rank k in complexity $\mathcal{O}\left(nnz(M) + \frac{nk^2}{\epsilon^4} + \frac{k^3}{\epsilon^5}\right)$. Just as a note here, when you see ϵ^4, ϵ^5, it's typically Woodruff and some other people. This is very good for warmstarts, since those don't require that good epsilons.

Bhojanapalli, Jain, Sanghavi get a spectral norm guarantee bound (opt_2 is optimal spectral norm) of

$$\|M - \tilde{M}\|_2 \leq \text{opt}_2 + \epsilon\|M\|_F$$

in time $\mathcal{O}\left(nnz(M) + \frac{nr^5(\lambda_1/\lambda_2)^2\log n}{\epsilon^2}\right)$.

Note you can use something like Frank-Wolfe to get an ϵ-approximation of this low-rank matrix approximation. It's comparable to alternating-minimization.

2.3 Matrix Completion Previous Results

There are more results for matrix completion $W_{ij} = 0, 1$.

Hazan Shai-Shalev result may apply to a full case: It says that the sample complexity for an ϵ- entrywise prediction behaves like $\frac{r*n^{1.5}}{\epsilon^2}$. I want to say that in weighted low-rank, you don't care about prediction: You care about recovery. For example, for word vectors, you don't care about predicting the joint probabilities. Recovery implies prediction, but not the other way around. Otherwise you can just do improper learning (see Hazan).

Candes and Plan get a $\mu^2rn\text{polylog}(n)$ number of samples, to get an error $\|M - \tilde{M}\|_F \leq n\|(M - M^*)_\Omega\|_F$. Note that M^* is optimal solution to the low-rank problem. This is just the Frobenius norm restricted to Ω, which is the samples. There is another guarantee from Keshavan et al, and optimization on the Grassmanian manifold and get $\mu rn\log n$ samples with $\|M - \tilde{M}\|_F \leq n\sqrt{k}\|(M - M^*)_\Omega\|_F$ guarantee. Moritz Hardt gets $(r^3/\epsilon^2n\mu\log n)$ samples (more samples) to get $\|M - \tilde{M}\|_F \leq \epsilon\|M\|_F$ error. Last year's FOCS, there is a 70-page paper by Sun and Luo which proves that with $n\max(r\mu\log n, \mu^2r^7)$: You can use SGD, GD and other methods for this. This result is very important. They have a very difficult lemma and combine it with our easy lemma, then we get weighted low-rank approximation. But they have very difficult geometry lemma.

Remark 2.1. Why is prediction not good enough? You have M already in word-embedding (it's the PMI). $\tilde{M}$ is **purer** than M - it purifies. Dimension reduction is the wrong metaphor: It denoises at the same time. But don't we think that in pure theory, we should have high dimension? We provide the vectors to find tasks. NLP doesn't even know the tasks to solve. At this point there are probably thousands of new tasks; this is the power of unsupervised learning. Even in all of Wikipedia, all 10^{10} pairs will not be there. They previously called the better performance of low-dimensional vectors “regularization”, but this is clearly wrong since generalization theory doesn’t even apply to unsupervised learning.

2.4 Weighted Low-Rank Previous Results

We achieve something like

$$\|M - \tilde{M}\|_2 \leq r\|W \odot [M - M^*]\|_2 + \epsilon$$

where $\odot$ is the Hadamard product, essentially allowing us to bound the spectral norm. This is r-optimal, but it is not $(1+\epsilon)$ optimal. We have $\mathbf{E}_{avg}[W_{ij}] = 1$ by initialization. We get runtime $\mathcal{O}\left(nnz(M)\dot{r}^2 \log(1/\epsilon)\right)$. Spectral norm will actually imply weighted Frobenius norm (you can optimize the ϵ etc). Then there is another Woodruff paper which appeared in STOC. Woodruff, Song et al got

$$\|M - \tilde{M}\|_W \leq (1+\epsilon)\|M - M^*\|_W$$

in time

$$\mathcal{O}\left(nnz(M) + nnz(W) + n2^{\frac{\text{rank}(\tilde{M})^3 \text{rank}(W)}{\epsilon}}\right)$$

with no assumptions on the matrix. Even the epsilon and both ranks are in the exponent. Actually, $\text{rank}(W)$ is the number of distinct columns. They hide some $+\delta$ and $+\log(1/\delta)$ in their optimization. Note that this is “fixed parameter tractable”.

3 The Technical Part

How do we minimize $F(x)$ with F nonconvex? Let’s call the minimizer x^*. Now apply gradient descent and alternating minimization, and the goal is to show that the algorithm works. What’s the condition of this function so that the algorithm can work? There are three types of ways to go about proving convergence for non-convergence:

1. Lyapunov type of argument. Basically you will define a divergence function $D(x,y)$ such that $D(x,x) = 0$, and $D(x,y) \geq 0$ for $x \neq y$. Then we want to show that $D(x_t, x^*)$ is decreasing over some gradient descent type of alternating minimization. (Moritz Hardt paper). We use this in our work.

2. The second method is show some kind of local convexity of the function. Show that $\forall x$, then $\langle \nabla F(x), x - x^* \rangle \geq \sigma\|x - x^*\|^2 + \eta\|\nabla F(x)\|^2$ (Tengyu-convexity). Here you can go to the optimum at a linear rate for a smooth function. If there is some bias you do’nt go directly towards the ground truth. All you probably need is the direction of motion. Basically, every direction of the gradient is correlated with the ground truth direction. In one-dimension this is quasi-convexity. If you have just direction of motion, it’s not clear that there are saddlepoints. (Dictionary Learning paper, Matrix Completion with Stochastic Gradient Descent).

3. The third case is $\|\nabla F(x)\|_2^2 \geq \sigma(F(x) - F(x^*))$. We’re basically saying it’s large. This is going to require smoothness. We always assume F is L-smooth. We only care about x^* here. This will also converge to ground truth. (Matrix Completion with Stochastic Gradient Descent). Perhaps this is not used in alternating minimization.

There are some things which satisfy the second one and not the third one, and not the third one but the second one. In the language of linear coupling, then the first two are “mirror descent” (with different regularizers), and the third condition is gradient descent.

We basically apply the first method for our result. We define a divergence function

$$\text{Div}(U, X) = \min_{Q \in D^{k \times k}, \sigma_{min}(Q) \geq 1/2, \sigma_{max}(Q) \leq 2} \|UQ - X\|_2$$

where σ are the singular values. Consider the orthogonal case: This is basically the subspace that this is. If U, V define the same subspace then there exists an orthogonal matrix Q such that $UQ = V$, so the distance is zero. So this is like approximate subspace distance between non-orthogonal matrices: We relax the spectral norm between $1/2$ and 2 instead. That is what our divergence is; we just show that we can always decrease it with our alternating minimization steps.

3.1 The Algorithm

We have $A_t, B_t \in \mathbb{R}^{n \times r}$. Then

$$B_{t+1} = \text{argmin}_B \|M - A_t B^T\|_W$$

If you fix A_t, it will have an explicit formula, so you can just solve that. Then you do

$$A_{t+1} = \text{argmin}_A \|M - AB_t\|_W$$

You also check if i^{th} row of $\|(A_{t+1})_i\|_2 \geq$ some constant, then you set $(A_{t+1})_i = 0$. If the row l_2 norm is too large, you just set it to 0. Given the divergence we defined before, we can show that it decreases over the course of this algorithm, proving optimization.

In the Moritz paper, there's a whitening step where you add Gaussian noise and so on. Here, we can run in linear time; our whitening step is very simple (setting the rows to 0). Throwing out these large rows give you a smaller Frobenius norm since our matrix is only rank r, however, spectral norm is also related to Frobenius norm by a factor of r or so given our bounds on the singular values.

So why does it work? We want to show $\text{Div}(A, U)$ is small. Basically, since we look at $U\Sigma$, $U\Sigma^{1/2}$ in bounding this, and we placed bounds on what the singular values could be, we are able to tolerate variation here and get decreasing result.