Sparse Large Margin Matrix Factorization*

Jason D. M. Rennie
jrennie@csail.mit.edu

March 11, 2005

Let Y, U, and V be $n \times m$ matrices. Y is binary; U and V are real. We want to update U and V so that $UV^T \approx Y$. We detail the alternating SVM algorithm described in (Srebro et al., 2005).

1 Learning V with fixed U

Assume U fixed. Each row of V can be solved for using the Support Vector Machine.

Define

- $\hat{I}(i,j)$ - number of observed entries in column j with row index $\leq i$.
- $\hat{J}(i,j)$ - number of observed entries in row i with column index $\leq j$.
- $I(s)$ - row index of observed entry $s \in S$
- $J(s)$ - column index of observed entry $s \in S$

Let j index a column of Y (and, correspondingly, a row of V). For all $s \in S$ such that $J(s) = j$, let

$$i = I(s), \tag{1}$$
$$\hat{i} = \hat{I}(i,j), \tag{2}$$
$$\vec{x}_{\hat{i}} = U_{i\cdot}, \text{ and} \tag{3}$$
$$y_{\hat{i}} = Y_{ij}. \tag{4}$$

We train an SVM on examples $\{\vec{x}_{\hat{i}}\}$ and labels $\{y_{\hat{i}}\}$. We set row j of V to the learned weight vector,

$$V_{j\cdot} = \vec{w}. \tag{5}$$

*Joint work with Nati Srebro and Tommi Jaakkola.

2 Learning U with fixed V

Assume V fixed. Each row of U can be solved for using the Support Vector Machine. Make the same definitions as above. Let i index a row of Y (and, correspondingly, a row of U). For all $s \in S$ such that $I(s) = i$, let

$$
\begin{aligned}
j &= J(s), & (6)\\
\hat{j} &= \hat{J}(i,j), & (7)\\
\vec{x}_{\hat{j}} &= V_{j\cdot}, \text{ and} & (8)\\
y_{\hat{j}} &= Y_{ij}. & (9)
\end{aligned}
$$

We train an SVM on examples $\{\vec{x}_{\hat{j}}\}$ and labels $\{y_{\hat{j}}\}$. We set row i of U to the learned weight vector,

$$U_{i\cdot} = \vec{w}. \tag{10}$$

3 Kernel Values

If Y is large, it may not be possible to store the entire U and V matrices. Here we discuss how to execute the algorithm using only kernel products. The dual objective of the SVM involves kernel products between pairs of examples.

Let j index a column of Y.

$$V_{j\cdot}^{(t+1)} = \sum_{s \in S | J(s)=j} \alpha_{\hat{i}} y_{\hat{i}} U_{I(s)\cdot}^{(t)}. \tag{11}$$

Let i index a row of Y.

$$U_{i\cdot}^{(t)} = \sum_{s \in S | J(s)=i} \alpha_{\hat{j}} y_{\hat{j}} V_{J(s)\cdot}^{(t)}. \tag{12}$$

Fixed U: Let A be matrix of learned $\{\vec{\alpha}\}$. Each row of A corresponds to a row of V. Zeros correspond to unobserved entries. Then $V = AU^T$.

Fixed V: Use β to denote dual parameters for this problem. Let B be matrix of learned $\{\vec{\beta}\}$. Each row of B corresponds to a row of U. Zeros correspond to unobserved entries. Then $U = BV^T$.

Use subscript to denote iteration:

$$
\begin{aligned}
V_{(t)} &= A_{(t)} U_{(t-1)}^T & (13)\\
U_{(t)} &= B_{(t)} V_{(t)}^T & (14)\\
U_{(t+1)} &= B_{(t)} U_{(t)} A_{(t)}^T & (15)\\
U_{(t+2)} &= B_{(t+1)} B_{(t)} U_{(t)} A_{(t)}^T A_{(t+1)}^T & (16)
\end{aligned}
$$

Note that A and B are sparse—only as many entries as number of observed items.

References

Shashua, A., & Levin, A. (2003). Ranking with large margin principle: Two approaches. *Advances in Neural Information Processing Systems 15*.

Srebro, N., Rennie, J. D. M., & Jaakkola, T. (2005). Large margin matrix factorization. *Advances in Neural Information Processing Systems 17*.