Neural Network Matrix Factorization

(paper written by: GK Dziugaite, DM Roy, 2015)

Dmitrii Meinster

NRU HSE, CS faculty, Data Science programme

20.12.2017

Matrix factorization problem (MF)

- Suppose we have some big matrix, $X \in \mathbb{R}^{N \times M}$, but only $X_{ij \in J \subset [1..N] \times [1..M]}$ are known.
- Want to find $U \in \mathbb{R}^{D \times N}$, $V \in \mathbb{R}^{D \times M}$, such that: $D \ll N, M; X \approx U^T V$.
- Possible applications: collaborative filtering, knowledge graphs

Probability matrix factorization (PMF)

(R. Salakhutdinov, A. Mnih, 2008)

- Assume $X_{ij} \sim \mathcal{N}([U^T V]_{ij}, \sigma^2)$
- Find U and V by minimizing $\|U^T V - X\|_F$
- Very effective in practice, but can be further improved
- BiasedPMF (Koren et. al., 2009): $X_{ij} \sim \mathcal{N}([U^T V]_{ij} + \mu_i + \tau_j + \beta, \sigma^2)$

PMF note

- Assume $f(w_1, w_2, \ldots) = \sum_j w_j$
- PMF finds mean of X_{ij}
  in form of $f(U_i \circ V_j)$ (elementwise product of D-dimensional vectors)
- Similary, in BiasedPMF, mean of X_{ij} is modeled by $f(U_i \circ V_j, \mu_i, \tau_j, \beta)$.
- What if we learn f instead of explicitly defining it?..

NNMF model

- Again, $X \in \mathbb{R}^{N \times M}$ with some unknown elements (J — set of known ones)
- To each row $n \in [1..N]$, associate latent feature vector $U_n \in \mathbb{R}^D$ and latent feature matrix $U'_n \in \mathbb{R}^{D' \times K}$
- Similarly, for each column $m \in [1..M]$, we have feature vector $V_m \in \mathbb{R}^D$ and latent feature matrix $V'_m \in \mathbb{R}^{D' \times K}$
- $U'_{n,i}$ — i'th row of matrix U'_n (K-dimensional vector); similar for V'_m.
- (U, V) — collection of all latent features
- Find $\hat{X}_{nm}$ in form of $f_\theta(U_n, V_m, U'_{n,1} \cdot V'^T_{m,1}, U'_{n,2} \cdot V'^T_{m,2}, \ldots)$, where f_θ is function computed by neural network with set of weights θ.

Learning

- Minimize objective:

$$\sum_{(n,m)\in J} (X_{nm} - \hat{X}_{nm})^2 + \lambda \left[\sum_n \| U'_n \|_F^2 + \sum_m \| V'_m \|_F^2 + \sum_n \| U_n \|_2^2 + \sum_m \| V_m \|_2^2 \right]$$

- On each step, alternate between optimizing neural network weights and the latent features;
- RMSProp was used.

Related work

- Random Function Model (RFM) (Lloyd et al., 2012): gaussian process instead of NN;
- Neural Tensor Network (NTN) (Socher et al., 2013): single-layer NN with third-order tensor instead of matrix; first layer of NNMF can be expressed in this form;
- Local Low Rank Matrix Approximation (LLORMA) (Lee et al., 2013): approximate each entry by (unique) combination of low-rank matrices;
- I-AutoRec (Sedhain et al., 2015): use autoencoder to generate full matrix from observed ratings; current state-of-the-art for such tasks.

Results

$K = 1, D' = 60, D = 10$, each hidden layer contained 50 (for 3HL) or 20 (for 4HL) units.

	NIPS	Protein	ML100k	ML1m
Vertices X	234	230	943	6040
Vertices Y	-	-	1682	3900
Edges	27144	52900	100000	1000209

Table 1: Data sets and their dimensions. The mark "-" highlights that the array is square.

	NIPS	Protein	ML-100K		ML-1M
RFM (3)	0.110	0.136	-	PMF (60)	0.883
PMF (3)	0.130	0.139	-	LLORMA-GLOBAL	0.865
PMF (60)	0.062	0.104	0.952	I-RBM	0.854
BiasedMF (60)	0.065	0.111	0.911	BiasedMF (60)	0.852
NTN (60)	0.048	0.071	0.910	NTN (60)	0.852
NNMF (3HL)	0.040	0.065	0.907	LLORMA-LOCAL	0.833
NNMF (4HL)	-	-	0.903	I-AutoRec	0.831
				NNMF (3HL)	0.846
				NNMF (4HL)	0.843

Table 2: Results across the four data sets for a variety of techniques. The token (D) specifies that a rank-D factorization was used. The token $(n\text{HL})$ specifies that n hidden layers were used. Scores reported for RFM and PMF (3) are taken from (Lloyd et al., 2012). Scores for BiasedMF were obtained using LibRec (Guo et al., 2015). Scores for LLORMA were taken from (Lee et al., 2013), AutoRec and RBM, were taken from (Sedhain et al., 2015).

Discussion

- NNMF beats latent feature models, but dominated by approaches which use local graph structure
- Further improvements are possible by optimizing NN structure

Thanks!

Questions?