DS-GA 1003 Project Proposal: Research Paper Recommendation

Jungkyu (JP) Park, Dae Young Kim, Yu-Hsuan Shih

1 Introduction

Our objective is to build a recommendation system for scholars by which we want to recommend papers that

1. scholars will find useful without need for extensive search.
2. scholars plan to write but are already written preventing redundancy.

We mainly plan on trying Network-based approaches where we will look at a directed citation network, treat missing data as zero, and predict what those values should be by collaborative filtering. The two matrices we will use are paper-by-paper matrix and scholar-by-paper matrix of which entries stand for references.

We would like to start with first approach as our baseline method. We will approximate the original matrix by a rank-k matrix, but it would be reasonable to build a model and minimize loss (in sense of Frobenius norm) over observed values using gradient descent, in a similar fashion to matrix factorization.

In terms of evaluations, we would like to observe diverse performance methods. Motivated by the absolute effectiveness of recommendation approaches. For instance, an algorithm that achieved a recall of 4% on an IEEE dataset achieved a recall of 12% on an ACM dataset. The relative effectiveness of two approaches is also not necessarily the same with different datasets. Hence, we will explore precision and recall, MRR, MAP, or nDCG, and we will compare different evaluation methods for different types of models that we build.

2 Data

We would like to use Aminer [4], a data set that extracted from DBLP, ACM, MAG (Microsoft Academic Graph), and other sources. The latest version contains 3,079,007 papers and 25,166,994 citation relationships. In the file, each line represents a paper, which is in JSON schema containing information of the paper's title, authors, venue, year, references and abstract.

3 Problem Definition

Let n be the total number of papers and p be the total number of scholars in our data set. In terms of Statistical Learning Theory, the possible input spaces $\mathcal{X}$ in our problem would be –

1. $\{0,1\}^n$, where an entry of a vector is either 0 or 1 indicating the citation relationship between two papers:

$$\mathbf{a}^i \in \mathcal{X} \text{ s.t. } \mathbf{a}^i_j = \begin{cases} 1, \text{Paper } i \text{ cites Paper } j \\ 0, \text{otherwise} \end{cases}$$

2. $\mathbf{N}^n$, where an entry of a matrix is a positive integer indicating how many times a scholar cites a paper:

$$\mathbf{b}^i \in \mathcal{X} \text{ s.t. } \mathbf{b}^i_j = \# \text{ of times scholar } i \text{ cites paper } j \ .$$

Note that in the input space $\mathcal{X}$, we treat 0 entries in the vectors as missing values. For the action space $\mathcal{A}$ and outcome space $\mathcal{Y}$, assume we are going to recommend k papers for each paper (or scholar), then both $\mathcal{A}$ and $\mathcal{Y}$ will be spaces contains k-dimensional vectors, i.e. $\mathcal{A} = \mathcal{Y} = \mathbf{N}^k$, where for $\hat{\mathbf{y}}^i \in \mathcal{A}$, $\hat{\mathbf{y}}^i_j,\ j = 0, \ldots, k$ is the j-th paper that we end up recommending to people interested in paper i (or to scholar i); and for $\mathbf{y}^i \in \mathcal{Y}$, $\mathbf{y}^i_j,\ j = 0, \ldots, k$ is a measurement of how paper i (or scholar i) is satisfied with the j-th paper that our recommendation system recommends. For discussion of algorithms below, define matrix $A \in \{0,1\}^{n\times n}$ whose rows are $\mathbf{a}^i,\ i = 1, \ldots, n$ and matrix $B \in \mathbf{N}^{n\times n}$ with rows $\mathbf{b}^i,\ i = 1, \ldots, p$.

4 Baseline Algorithms

4.1 Popularity model

We create a list of papers in decreasing number of being cited, and recommend the same list of papers to everyone. Although this method sounds naive, doing this may help us detect the popularity bias in other models.

4.2 Matrix Factorization Model in Collaborative Filtering Algorithms

One of the most common algorithms for recommendation system is Collaborative Filtering (CF). In terms of paper recommendation, it is based on an assumption that papers (scholars) that have similar citation pattern might be more likely to have similar reactions on other papers than a random paper (scholar) . Considering the sparsity of our design matrices A and B (each paper usually cites no more than 50 papers while we have more than 1 million papers in our data set), we decide to start from experimenting Matrix Factorization models in CF algorithms for our recommendation system.

Matrix Factorization models map both users (papers or scholars in our problem) and items (papers) to a joint latent factor space of dimensionality f, i.e. user i is associated with $p_i \in \mathbf{R}^f$ and item j is associated with $q_j \in \mathbf{R}^f$, and model the user-item interactions by the inner product in that space, i.e. $p_i^T q_j$. If we explain q_j as a vector that characterizes item j by f different characteristics, and p_j as a vector of how user i reacts with those f characteristics, then $p_i^T q_j$ can be explained as a number modeling user i's rating (citations) of item j. Therefore, our objective can then be formed as following optimization problem, where A is defined in section 3 and ℓ_2 loss function is used –

$$\min_{p_i, q_j \in \mathbf{R}^f} \sum_j \sum_i (A_{i,j} - p_i^T q_j)^2 \tag{1}$$

Let P be a matrix with columns p_i and Q be a matrix with columns q_i, a matrix expression of (1) is

$$\min_{P,Q \in \mathbf{R}^{n \times f}} \|A - P^T Q\|_F^2 \tag{2}$$

By singular value decomposition (SVD), $A_k = U_k \Sigma_k V_k^T = \arg\min_{\text{rank}(B)=k} \|A - B\|_F$, where $A = U \Sigma V^T$ and $U_k = U[:, 1:k]$, $\Sigma_k = \Sigma[1:k, 1:k]$ and $V_k = [1:k, :]$ (in matlab matrix expression). Therefore, if we have SVD of A, the optimal value of (2) can be achieved by $P^* = \sqrt{\Sigma_k} U_k^T$, $Q^* = \sqrt{\Sigma_k} V_k^T$, where $(P^*)^T Q^* = U_k \sqrt{\Sigma_k} \sqrt{\Sigma_k} V_k^T = U_k \Sigma_k V_k^T = A_k$.

While we already find an analytical solution of (2), when A is large, SVD of A can be computationally unachievable. One way to avoid calculating SVD of A for solving (2) is by using method of gradient descent with objective function $F(P,Q) = \|A - P^T Q\|_F^2 = \text{tr}\left((A - P^T Q)(A - P^T Q)^T\right)$, where the gradient of F is given by

$$\nabla_{\mathbf{P}} F(P,Q) = -2AQ^T + 2P^T Q Q^T$$

and

$$\nabla_{\mathbf{Q}} F(P,Q) = -2AP^T + 2P^T Q P^T.$$

This leads to our first baseline algorithm 1 –

Algorithm 1 Gradient descent method for objective function $F(P,Q) = \|A - P^T Q\|_F^2$

```
Given $P, Q \in \mathbf{R}^{n \times f}$, n number of papers, dimension of latent factor space $f \in \mathbf{N}$, step size $\eta \in \mathbf{R}$
repeat
    1. $\Delta P := -\nabla_P F(P,Q)$, $\Delta Q := -\nabla_Q F(P,Q)$.
    2. Update: $P := P + \eta \Delta P$, $P := P + \eta \Delta Q$
until stopping criterion is satisfied.
```

4.3 How to recommend using baseline models?

Let k be the number of papers that we are going to recommend. To do the recommendation basing on the result we get from baseline models to paper (scholar) i, we can separate the situation into two cases:

1. Fix amount of recommendation, i.e. k is a constant –
   $\Rightarrow$ Recommend those papers with k highest values in the i-th row of approximated matrix P^TQ.

2. The amount of recommendation varies over users, i.e. k is a variable –
   $\Rightarrow$ Recommend all the papers with value larger than a given ϵ in the i-th row of P^TQ.

Both of the cases lead to the question of which k or ϵ we should use. Similar to choosing penalty parameter λ in the ridge regression model, we can first decide a performance evaluation metric discussed in section 5, then do hyper-parameter search on k (or ϵ) to find k (or ϵ) such that our baseline model with that k (or ϵ) performs the best in term of the performance evaluation metric.

5 Performance Evaluation

Since we don't have live users, we will focus on the offline evaluation of our recommendation system.

Offline evaluation metrics can be categorized into (1) accuracy metrics and (2) discovery-oriented metrics. Accuracy metrics focus on evaluating how accurate the recommendations are; discovery-oriented metrics measure the usefulness of recommendation system. Since we want our recommendations to match the users preference and be useful, we would like to use both of them for the performance evaluation.

5.1 Accuracy metrics

- Precision@k checks whether the original citing papers were ranked high for the query manuscript. This is motivated by the notion that people are likely to cite papers that are highly correlated to their own literature up on their list.

- Recall@k is defined as the percentage of original citing papers that appear in the top-k recommended list.

- MAP (Mean Average Precision): the average of the precision value obtained for the set of top k documents existing after each relevant document is retrieved, and this value is then averaged over information needs. `https://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-ranked-retrieval-results-1.html`

- nDCG (normalized discounted cumulative gain): is designed for situations of non-binary notions of relevance. Like precision atk, it is evaluated over some numberkof top search results.

- MRR: the inverse of the rank at which the removed paper was recommended.

5.2 Discovery-oriented metrics

- Diversity metrics: metrics that measure whether the system can recommend various types of items to users. e.g. gini coefficient, temporal diversity, intra-list similarity, MMR, etc

- Novelty metrics: metrics that measure whether the recommended items are unknown to the user, e.g. discovery ratio, precision of novelty, long tail metric, generalized novelty model, etc

- Serendipity metrics: metrics that measure whether the recommendation gives the surprise to the user, e.g. unexpectedness, entropy-based diversity, unserendipity, etc

More details on how to calculate these metrics - `http://soc-research.org/wp-content/uploads/2014/11/OfflineTest4RS.pdf`

6 List of ML Models for Recommendation System

The following models use MSE loss and have different bias, regularization terms, and candidate domain.

- **Funk's SVD**

- **SVD++**

- **Koren's SVD++**
- **Limit candidate set** Limit domain of papers to calculate the loss [1]. This model can recommend into test + randomly selected unknown items. Good-but-unknown items are probably not in the candidate set, so we do not want to penalize potentially good recommendations due to missing data. However, this approach might be prone to popularity bias [2].

The following models learn to rank recommendations. By only measuring rank effectiveness with user-paper matrix, we don't penalize missing entries and hope to get better recommendation.

- **GRank, Shams and Haratizadeh** (2016) [3]
- **Large-scale Collaborative Ranking in Near-Linear Time (Wu et al.)** [5]

7 Time Line

Date	Plan
March 18th - Marth 31st	Finish running baseline algorithms on all data sets
April 1st - April 7th	Try Funk's SVD, SVD++, Koren's SVD++
April 8th - April 14th	Try Limit candidate set with all the algorithms above
April 15th - April 21st	Write-up result and decide what to do next
April 22th - May 5th	Try models with ranking based loss
May 6th - May 11th	Writing-up final report

References

[1] Yehuda Koren. "Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model". In: *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '08. Las Vegas, Nevada, USA: ACM, 2008, pp. 426–434. ISBN: 978-1-60558-193-4. DOI: 10.1145/1401890.1401944. URL: http://doi.acm.org/10.1145/1401890.1401944.

[2] Vaibhav Mahant. "Improving Top-n Evaluation of Recommender Systems". PhD thesis. 2016.

[3] Bita Shams and Saman Haratizadeh. "Graph-based collaborative ranking". In: *Expert Systems with Applications* 67 (2017), pp. 59–70.

[4] Jie Tang et al. "ArnetMiner: Extraction and Mining of Academic Social Networks". In: *KDD'08*. 2008, pp. 990–998.

[5] Liwei Wu, Cho-Jui Hsieh, and James Sharpnack. "Large-scale Collaborative Ranking in Near-Linear Time". In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2017, pp. 515–524.