Factor Models for Tag Recommendation in BibSonomy

Steffen Rendle and Lars Schmidt-Thieme

Machine Learning Lab, University of Hildesheim, Germany
{srendle,schmidt-thieme}@ismll.uni-hildesheim.de

Abstract. This paper describes our approach to the ECML/PKDD Discovery Challenge 2009. Our approach is a pure statistical model taking no content information into account. It tries to find latent interactions between users, items and tags by factorizing the observed tagging data. The factorization model is learned by the Bayesian Personal Ranking method (BPR) which is inspired by a Bayesian analysis of personalized ranking with missing data. To prevent overfitting, we ensemble the models over several iterations and hyperparameters. Finally, we enhance the top-n lists by estimating how many tags to recommend.

1 Introduction

In this paper, we describe our approach to task 2 of the ECML/PKDD Discovery Challenge 2009. The setting of the challenge is personalized tag recommendation [1]. An example is a social bookmark site where a user wants to tag one of his bookmark and the tag recommender suggest the user a personalized list of tags he might want to use for this item.

Our approach to this problem is a pure statistical model using no content information. It relies on a factor model related to [2] where the model parameters are optimized for the maximum likelihood estimator for personalized pairwise ranking [3]. Furthermore, we use a smoothing method for reducing the variance in the factor models. Finally, we provide a method for estimating how many tags should be recommended for a given post. This method is model independent and can be applied to any tag recommender.

2 Terminology and Formalization

We follow the terminology of [2]: U is the set of all users, I the set of all items/ resources and T the set of all tags. The tagging information of the past is represented as the ternary relation $S \subseteq U \times I \times T$. A tagging triple $(u, i, t) \in S$ means that user u has tagged an item i with the tag t. The *posts* P_S denotes the set of all distinct user/ item combinations in S:

$$P_S := \{(u, i) | \exists t \in T : (u, i, t) \in S\}$$

Our models calculate an estimator $\hat{Y}$ for S. Given such a predictor $\hat{Y}$ the list Top of the N highest scoring items for a given user u and an item i can be calculated by:

$$\text{Top}(u, i, N) := \underset{t \in T}{\overset{N}{\text{argmax}}}\, \hat{y}_{u,i,t} \tag{1}$$

where the superscript N denotes the number of tags to return. Besides $\hat{y}_{u,i,t}$ we also use the notation of a rank $\hat{r}_{u,i,t}$ which is the position of t in a post (u, i) after sorting all tags by $\hat{y}_{u,i,t}$:

$$\hat{r}_{u,i,t} := |\{t' : \hat{y}_{u,i,t'} > \hat{y}_{u,i,t}\}|$$

3 Factor Model

Our factorization model (FM) captures the interactions between users and tags as well as between items and tags. The model equation is given by:

$$\hat{y}_{u,i,t} = \sum_f \hat{u}_{u,f} \cdot \hat{t}^U_{t,f} + \sum_f \hat{i}_{i,f} \cdot \hat{t}^I_{t,f} \tag{2}$$

Where $\hat{U}$, $\hat{I}$, $\hat{T}^U$ and $\hat{T}^I$ are feature matrices capturing the latent interactions. They have the following types:

$$\hat{U} \in \mathbb{R}^{|U| \times k}, \quad \hat{I} \in \mathbb{R}^{|I| \times k},$$
$$\hat{T}^U \in \mathbb{R}^{|T| \times k}, \quad \hat{T}^I \in \mathbb{R}^{|T| \times k}$$

Note that this model differs from the factorization model in [2] where the model equation is the Tucker Decomposition.

3.1 Optimization Criterion

Our optimization criterion is an adaption of the BPR criterion (Bayesian Personalized Ranking) [3]. The criterion presented in [3] is derived for the task of item recommendation. Adapted to tag recommendation, the optimization function for our factor model is:

$$\text{BPR-Opt} := \sum_{(u,i) \in P_S} \sum_{t^+ \in T^+_{u,i}} \sum_{t^- \in T^-_{u,i}} \ln \sigma(\hat{y}_{u,i,t^+} - \hat{y}_{u,i,t^-}) - \lambda(||\hat{U}||^2 + ||\hat{I}||^2 + ||\hat{T}^U||^2 + ||\hat{T}^I||^2) \tag{3}$$

That means BPR-Opt tries to optimize the pairwise classification accuracy within observed posts. Note that it differs from [2] by optimizing for pairwise classification (log-sigmoid) instead of AUC (sigmoid).

3.2 Learning

The model is learned by the LearnBPR algorithm [3] which is a stochastic gradient descent algorithm where cases are sampled by bootstrapping. In the following, we show how we apply this generic algorithm to the task of optimzing our model paramaters for the task of tag recommendation. The gradients of our model equation (2) with respect to the model parameters $\Theta = \{\hat{U}, \hat{I}, \hat{T}^U, \hat{T}^I\}$ are:

$$\begin{aligned}
&\frac{\partial \text{BPR-Opt}}{\partial \Theta} \\
&\quad = \sum_{(u,i) \in P_S} \sum_{t^+ \in T^+_{u,i}} \sum_{t^- \in T^-_{u,i}} \frac{\partial}{\partial \Theta} \ln \sigma(\hat{y}_{u,i,t^+} - \hat{y}_{u,i,t^-}) - \lambda \frac{\partial}{\partial \Theta} ||\Theta||^2 \\
&\propto \sum_{(u,i) \in P_S} \sum_{t^+ \in T^+_{u,i}} \sum_{t^- \in T^-_{u,i}} \frac{-e^{-(\hat{y}_{u,i,t^+} - \hat{y}_{u,i,t^-})}}{1 + e^{-(\hat{y}_{u,i,t^+} - \hat{y}_{u,i,t^-})}} \cdot \frac{\partial}{\partial \Theta} (\hat{y}_{u,i,t^+} - \hat{y}_{u,i,t^-}) - \lambda \Theta
\end{aligned}$$

That means, we only have to compute the derivations of our model equation $\hat{y}_{u,i,t}$ with respect to each model parameter from $\Theta = \{\hat{U}, \hat{I}, \hat{T}^U, \hat{T}^I\}$:

$$\begin{aligned}
\frac{\partial}{\partial \hat{u}_{u,f}} \hat{y}_{u,i,t} &= \hat{t}^U_{t,f} \\
\frac{\partial}{\partial \hat{i}_{u,f}} \hat{y}_{u,i,t} &= \hat{t}^I_{t,f} \\
\frac{\partial}{\partial \hat{t}^U_{t,f}} \hat{y}_{u,i,t} &= \hat{u}_{u,f} \\
\frac{\partial}{\partial \hat{t}^I_{t,f}} \hat{y}_{u,i,t} &= \hat{i}_{i,f}
\end{aligned}$$

These derivations are used in the stochastic gradient descent algorithm shown in figure 1.

The method presented so far has the following hyperparameters:

- $\alpha \in \mathbb{R}^+$ learning rate
- $\lambda \in \mathbb{R}^+_0$ regularization parameter
- $\mu \in \mathbb{R}$ mean value for initialization of model parameters
- $\sigma^2 \in \mathbb{R}^+_0$ standard deviation for initialization of model parameters
- $k \in \mathbb{N}^+$ feature dimensionality of factorization

Reasonable values for all parameters can be searched on a holdout set. The learning rate and the initialization parameters are only important for the learning algorithm but are not part of the optimization criterion or model equation. Usually, the values found for α, μ, σ^2 on the holdout generalize well.

In contrast to this, the regularization and dimensionality are more important for the prediction quality. In general, when the regularization is chosen properly, the higher the dimensionality the better. In our submitted result, we use an ensemble of models with different regularization and dimensionality.

procedure LEARNBPR($P_S, \hat{U}, \hat{I}, \hat{T}^U, \hat{T}^I$)
  draw $\hat{U}, \hat{I}, \hat{T}^U, \hat{T}^I$ from $N(\mu, \sigma^2)$
  repeat
    draw (u, i, t^+, t^-) uniformly from $P_S \times T^+_{u,i} \times T^-_{u,i}$
    $d \leftarrow \hat{y}_{u,i,t^+} - \hat{y}_{u,i,t^-}$
    for $f \in 1, \ldots, k$ **do**
      $\hat{u}_{u,f} \leftarrow \hat{u}_{u,f} + \alpha \left(\frac{e^{-d}}{1+e^{-d}} \cdot (\hat{t}^U_{t^+,f} - \hat{t}^U_{t^-,f}) + \lambda \cdot \hat{u}_{u,f} \right)$
      $\hat{i}_{i,f} \leftarrow \hat{i}_{i,f} + \alpha \left(\frac{e^{-d}}{1+e^{-d}} \cdot (\hat{t}^I_{t^+,f} - \hat{t}^I_{t^-,f}) + \lambda \cdot \hat{i}_{i,f} \right)$
      $\hat{t}^U_{t^+,f} \leftarrow \hat{t}^U_{t^+,f} + \alpha \left(\frac{e^{-d}}{1+e^{-d}} \cdot \hat{u}_{u,f} + \lambda \cdot \hat{t}^U_{t^+,f} \right)$
      $\hat{t}^U_{t^-,f} \leftarrow \hat{t}^U_{t^-,f} + \alpha \left(\frac{e^{-d}}{1+e^{-d}} \cdot -\hat{u}_{u,f} + \lambda \cdot \hat{t}^U_{t^-,f} \right)$
      $\hat{t}^I_{t^+,f} \leftarrow \hat{t}^I_{t^+,f} + \alpha \left(\frac{e^{-d}}{1+e^{-d}} \cdot \hat{i}_{i,f} + \lambda \cdot \hat{t}^I_{t^+,f} \right)$
      $\hat{t}^I_{t^-,f} \leftarrow \hat{t}^I_{t^-,f} + \alpha \left(\frac{e^{-d}}{1+e^{-d}} \cdot -\hat{i}_{i,f} + \lambda \cdot \hat{t}^I_{t^-,f} \right)$
    end for
  until convergence
  return $\hat{U}, \hat{I}, \hat{T^U}, \hat{T^I}$
end procedure

Fig. 1. Optimizing our factor model for equation (3) with bootstrapping based stochastic gradient descent. With learning rate α and regularization λ.

3.3 Ensembling Factor Models

Ensembling factor models with different regularization and dimensionality is supposed to remove variance from the ranking estimates. There are basically two simple approaches of ensembling predictions $\hat{y}^l_{u,i,t}$ of l models:

1. Ensemble of the **value estimates** $\hat{y}^l_{u,i,t}$:

$$\hat{y}^{\text{ev}}_{u,i,t} := \sum_l w_l \cdot \hat{y}^l_{u,i,t} \tag{4}$$

2. Ensemble of the **rank estimates** $\hat{r}^l_{u,i,t}$:

$$\hat{y}^{\text{er}}_{u,i,t} := \sum_l w_l \cdot (|T| - \hat{r}^l_{u,i,t}) \tag{5}$$

   That means tags with a high rank (low $\hat{r}$) will get a high score $\hat{y}$.

Where w_l is the weighting parameter for each model.

Whereas ensembling value estimates is effective for models with predictions on the same scale, rank estimates are favorable in cases where the $\hat{y}$ values of the different models have no direct relationship.

Ensembling Different Factor Models For our factor models the scales of $\hat{y}$ depend both on the dimensionality and the regularization parameter. Thus we

use the rank estimates for ensembling factor models with different dimensionality and regularization. In our approach we use a dimensionality of $k \in \{64, 128, 256\}$ and regularization of $\lambda \in \{10^{-4}, 5 \cdot 10^{-5}\}$. As the prediction quality of all of our factor models are comparable, we have chosen identical weights $w_l = 1$.

Ensembling Iterations Within each factor model we use a second ensembling strategy to remove variance. Besides the hyperparameters, another problem is the stopping criterion of the learning algorithm (see figure 1). We stop after a predefined number of iterations (2000) – we have chosen an iteration size of $10 \cdot |S|$ single draws. In our experiments the models usually converged already after about 500 iterations but in the following iterations the ranking alternates still a little bit. To remove the variance, we create many value estimates from different iterations and ensemble them. I.e. after the first 500 iterations we create each 50 iterations a value estimate for each tag in all test posts and ensemble these estimates with (4). Again there is no reason to favor an iteration over another, so we use identical weights $w_l = 1$. This gives the final estimates for each model. The models with different dimensionality and regularization are ensembled as described above.

4 Baseline Models

Besides our factorization model we also consider several baseline models and ensembles of these models. The models we pick as baselines are *most-popular by item* (mpi), *most-popular by user* (mpu), *item-based knn* (knni) and *user-based knn* (knnu).

The most-popular models are defined as follows:

$$\hat{y}^{\text{mpi}}_{u,i,t} = |\{u' \in U : (u', i, t)\}|$$
$$\hat{y}^{\text{mpu}}_{u,i,t} = |\{i' \in I : (u, i', t)\}|$$

The k-nearest-neighbour models (knn) are defined as follows:

$$\hat{y}^{\text{knni}}_{u,i,t} = \sum_{(u,i',t) \in S} \text{sim}_{i,i'}$$
$$\hat{y}^{\text{knnu}}_{u,i,t} = \sum_{(u',i,t) \in S} \text{sim}_{u,u'}$$

To measure $\text{sim}_{i,i'}$ and $\text{sim}_{u,u'}$ respectively, we first fold/ project the observed data tensor in a two dimensional matrix F^U and F^I:

$$f^I_{i,t} = |\{u : (u, i, t) \in S\}|$$
$$f^U_{u,t} = |\{i : (u, i, t) \in S\}|$$

After the folding we apply cosine similarity to compare two tag vectors:

$$\mathrm{sim}_{i,i'} = \frac{\sum_t f^I_{i,t} \cdot f^I_{i',t}}{\sqrt{\sum_t (f^I_{i,t})^2} \cdot \sqrt{\sum_t (f^I_{i,t})^2}}$$

$$\mathrm{sim}_{u,u'} = \frac{\sum_t f^U_{u,t} \cdot f^U_{u',t}}{\sqrt{\sum_t (f^U_{u,t})^2} \cdot \sqrt{\sum_t (f^U_{u,t})^2}}$$

We tried different weighted ensembles of the baseline models using the value estimate ensembling method. Even though these ensembles produce quite good results, in our experiments they did not outperform the factor models and furthermore adding baselines to the factor models did not result in a significant improvement of the factor models. Thus our final submission only consists of the factor models.

5 Adaptive List Length

In contrast to the usual evaluation scheme of tag recommendation, in this challenge the recommender was free to choose the length of the list of the recommendations in a range from a length of 0 to 5. The evaluation functions are:

$$\mathrm{Prec}(S_{test}) := \operatorname*{avg}_{(u,i)\in P_{S_{test}}} \frac{|\operatorname{Top}(u,i,\min(5,\#_{u,i})) \cap \{t|(u,i,t) \in S_{test}\}|}{\min(5,\#_{u,i})}$$

$$\mathrm{Recall}(S_{test}) := \operatorname*{avg}_{(u,i)\in P_{S_{test}}} \frac{|\operatorname{Top}(u,i,\min(5,\#_{u,i})) \cap \{t|(u,i,t) \in S_{test}\}|}{|\{t|(u,i,t) \in S_{test}\}|}$$

$$\mathrm{F1}(S_{test}) := \frac{2 \cdot \mathrm{Prec}(S_{test}) \cdot \mathrm{Recall}(S_{test})}{\mathrm{Prec}(S_{test}) + \mathrm{Recall}(S_{test})}$$

Where $\#_{u,i}$ is the number of tags the recommender estimates for a post.

There are three simple ways to estimate $\#_{u,i}$:

- **Global estimate**:

$$\#^G_{u,i} := \frac{|S|}{|P_S|}$$

- **User estimate**:

$$\#^U_{u,i} := \frac{|\{(i',t) : (u,i',t) \in S\}|}{|\{i' : (u,i',t) \in S\}|}$$

- **Item estimate**:

$$\#^I_{u,i} := \frac{|\{(u',t) : (u',i,t) \in S\}|}{|\{u' : (u',i,t) \in S\}|}$$

Based on these simple estimators, a combined post size can be produced by a linear combination:

$$\#^E_{u,i} := \beta_0 + \beta_G \#^G_{u,i} + \beta_U \#^U_{u,i} + \beta_I \#^I_{u,i}$$

In our approach we use $\#^E_{u,i}$ and optimize β on the holdout set for maximal F1. We found that choosing an adaptive length of the recommender list significantly improved the results over a fixed number.

6 Experimental Results

6.1 Sampling of Holdout Set

As the test of the challenge was released two days before the submission deadline, we tried to generate representative holdout-sets. We created two test sets, one following the leave-one-post-per-user-out protocol [1] and a second one by uniformly sampling posts with the constraint that the dataset should remain a 2-core after moving a post into the test set. These two sets were used as holdout sets for algorithm evaluation and hyperparameter selection. In the following, we report results for the second holdout set, because its characteristics (in terms of number of users, items and posts) are closer to the real test set.

6.2 Results

The results of the method presented so far can be found in table 2 and 3. As you can see, the single baseline models result in low quality but ensembles can achieve a good quality. In contrast to this, our proposed factor models generate better recommendations. The best possible ensemble (optimized on test!) of the baselines achieves a score of 0.330 on the challenge set whereas our factor ensemble (not optimized on test) results in 0.345.

	mpu	mpi	mp-ens	knni	knnu	knn-ens	knn+mp-ens
holdout	0.249	0.351	-/0.423	0.401	0.371	-/0.445	-/0.473
challenge	0.098	0.288	0.290/0.317	0.209	0.295	0.293/0.320	0.299/0.330

Fig. 2. F-Measure quality for the baselines methods. For the ensembles, we report two results: one for an ensemble with identical weights and one with optimal weights that have been optimized on the test! set. For sure this is an optimistic value that might not be found using the holdout split.

An interesting finding is that the results on the challenge test set largely differs from both of our holdout sets. But as all methods suffer, we assume that the tagging behavior in the challenge test set is indeed different from the one in the training set. Especially, the baseline most-popular-by-user dropped largely from 24.9% to 9.8% – this might indicate that personalization is difficult to achieve on

	single FM	FM-ens	FM-ens adaptive list length
holdout	0.495 ± 0.002	0.498	0.522
challenge	-	0.345	**0.356**

Fig. 3. F-Measure quality for the factorization methods. Single FM reports the average quality of each factorization model. FM-ens is the unweighted ensemble and finally we report the ensemble with the adaptive list length, i.e. predicting sometimes less than 5 tags.

the challenge test set using the provided training set. Non-personalized methods or content-based methods could benefit from the difference in both sets. Also methods that can handle temporal changes in the tagging behaviour might improve the scores.

7 Conclusion

In this paper, we have presented a factor model for the task of tag recommendation. The model tries to describe the individual tagging behavior by four low-dimensional matrices. The model parameters are optimized for the personalized ranking criterion BPR-Opt [3]. The length of the recommended lists is adapted both to the user and item. Our evaluation indicates that our approach outperforms ensembles of baseline models which are known to give high quality recommendations [1].

8 Acknowledgements

The authors gratefully acknowledge the partial co-funding of their work through the European Commission FP7 project MyMedia (www.mymediaproject.org) under the grant agreement no. 215006. For your inquiries please contact info@mymediaproject.org.

References

1. Jaeschke, R., Marinho, L., Hotho, A., Schmidt-Thieme, L., Stumme, G.: Tag recommendations in social bookmarking systems. AICOM (2008)
2. Rendle, S., Marinho, L.B., Nanopoulos, A., Thieme, L.S.: Learning optimal ranking with tensor factorization for tag recommendation. In: KDD '09: Proceeding of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, New York, NY, USA, ACM (2009)
3. Rendle, S., Freudenthaler, C., Gantner, Z., Thieme, L.S.: Bpr: Bayesian personalized ranking from implicit feedback. In: Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI 2009). (2009)