Hybrid Recommendations by Content-Aligned Bayesian Personalized Ranking

Ladislav Peska

Department of Software Engineering

Faculty of Mathematics and Physics, Charles University, Prague

Czech Republic

peska@ksi.mff.cuni.cz

Hybrid Recommendations by Content-Aligned Bayesian Personalized Ranking

In many application domains of recommender systems, content-based (CB) information are available for users, objects or both. CB information plays an important role in the process of recommendation, especially in cold-start scenarios, where the volume of feedback data is low. However, CB information may come from several, possibly external, sources varying in reliability, coverage or relevance to the recommending task. Therefore, each content source or attribute possess a different level of informativeness, which should be taken into consideration during the process of recommendation.

In this paper, we propose a *Content-Aligned Bayesian Personalized Ranking Matrix Factorization* method (CABPR), extending Bayesian Personalized Ranking Matrix Factorization (BPR) by incorporating multiple sources of content information into the BPR's optimization procedure. The working principle of CABPR is to calculate *user*-to-*user* and *object*-to-*object* similarity matrices based on the content information and penalize differences in latent factors of closely related users' or objects'. CABPR further estimates relevance of similarity matrices as a part of the optimization procedure. CABPR method is a significant extension of a previously published BPR_MCA method, featuring additional variants of optimization criterion and improved optimization procedure.

Four variants of CABPR were evaluated on two publicly available datasets: *MovieLens 1M* dataset, extended by data from IMDB, DBTropes and ZIP code statistics and *LOD-RecSys* dataset extended by the information available from DBPedia. Experiments shown that CABPR significantly improves over standard BPR as well as BPR_MCA method w.r.t. several cold-start scenarios.

Keywords: Hybrid recommender systems; Bayesian personalized recommendation; Content alignment; Cold-start problem

1 Introduction

Recommender systems (RS) belongs to the class of automated content-processing tools, aiming to provide users with unknown, surprising, yet relevant objects without the necessity of explicitly query for them. As such, RS are complementary to both keyword

and attribute-based search engines, because recommendations can be provided implicitly, without user's conscious cooperation or detailed knowledge of his/her intentions.

The core of recommender systems are machine learning algorithms applied on the matrix of user to object past preferences. So far, the majority of research effort was spent on the collaborative filtering (CF) algorithms, i.e., recommendations based on other users' preference, while the preference matrix was derived from explicit feedback, e.g., user rating. This research direction was supported by some meta-analysis too, e.g., (de Campos, Fernández-Luna, Huete, & Rueda-Morales, 2010; Pilászy & Tikk, 2009) claimed that collaborative recommender systems are more accurate than content-based (CB) algorithms, i.e., recommendations based on CB similarity of objects or users, if a sufficient amount of user feedback is available.

State-of-the-art methods of collaborative filtering are mostly based on matrix factorization (MF) or their multi-layered deep learning extensions. Various (MF) approaches were proposed so far, e.g., SVD++ (Koren, Bell, & Volinsky, 2009), PMF (Salakhutdinov & Mnih, 2007), BPR (Rendle, Freudenthaler, Gantner, & Schmidt-Thieme, 2009), Logistic MF (Johnson, 2014) etc. The core idea of applying matrix factorization techniques in recommender systems is to map both users and objects into a shared low-dimensional latent feature space and use this representation to calculate the predicted relevance of any user-object pair. Matrix factorization techniques differ from one another especially in the optimization criteria, optimization procedure and the exact calculation of user-object score.

Explicit user ratings are rich and reliable source of user preferences, however they are often not numerous enough or even completely unavailable on some application domains (Hu, Koren, & Volinsky, 2008; L. Peska & Vojtas, 2017). Therefore,

recommender systems in praxis mostly rely on the usage of *implicit feedback*[1], i.e., processing user's actions committed during normal browsing as a source of user's preferences. Although there is some recent research on diversification of *implicit feedback* (e.g, Lerche & Jannach, 2014; Peska & Vojtas, 2017; Yi, Hong, Zhong, Liu, & Rajan, 2014), positive-only unary feedback is often the only available information.

Several MF techniques were designed to handle positive-only implicit feedback (also referred as one-class matrix factorization). One of the first such approaches were WRMF (Hu et al., 2008), Maximum Margin MF (Weimer, Karatzoglou, & Smola, 2008) or Bayesian Personalized Ranking MF (BPR, Rendle et al., 2009).

BPR was designed to optimize AUC-like ranking criterion with positive-only feedback information. Unlike other MF approaches, e.g., SVD++ or WRMF, BPR uses pairwise preference relations between known positive and unknown objects and therefore optimizes correct ranking of objects directly, instead of minimizing some rating prediction error as in SVD++ or WRMF.

So far, all mentioned MF methods were purely collaborative, i.e., using only preference matrix to calculate recommendations. However, the performance of purely collaborative methods drops significantly together with the decrease of available user feedback. This often happens during the early RS deployment phase and is usually referred as the *cold start problem* (Kluver & Konstan, 2014). In order to overcome the cold-start problem, recommender systems may rely on content-based attributes of users or objects and provide recommendations based on the similarity of CB information (Lops, de Gemmis, & Semeraro, 2011). Also, hybrid techniques combining CB and CF

---

[1] Domonkos Tikk: Lessons Learnt at Building Recommendation Services in Industry Scale, Industrial Keynote, ECIR 2016, *slideshare.net/domonkostikk/lessons-learnt-at-building-recommendation-services-at-industry-scale*

approaches are plausible. Several hybrid techniques using CB information in MF models were proposed recently. In Content-boosted MF, objects are defined through the latent factors of its binarized CB attributes (Forbes & Zhu, 2011). Several proposals extend SVD++ by imposing similarity of latent factors based on the CB similarity of objects or users (Nguyen & Zhu, 2013; Zhen, Li, & Yeung, 2009).

Another related approach, MSCMF (Zheng, Ding, Mamitsuka, & Zhu, 2013), is an extension of WRMF method and was originally proposed for drug-target interaction prediction problem, however, it can be applied on the recommender systems domain as well.

In our previous work (Ladislav Peska, 2017), we focused on a content-based extension of BPR and proposed BPR_MCA method. BPR_MCA extends BPR's optimization criterion by requiring similarity of closely related users' (objects') latent factors w.r.t. arbitrary many similarity matrices, while the relevance of similarity matrices is estimated by the optimization procedure as well. This paper is an extension of our previous work on BPR_MCA, however the proposed *Content-Aligned Bayesian Personalized Ranking Matrix Factorization* method (CABPR) differs significantly from BPR_MCA in two aspects. First, CABPR features several additional variants of incorporating CB similarity into the BPR's optimization criterion (see Section 3.2). Second, an improved optimization procedure was proposed and implemented. Furthermore, we compare the results of CABPR and BPR_MCA on one additional dataset; CABPR algorithm outperformed BPR_MCA in each of the evaluated scenarios (see Table 6).

Our approach also extends the work of Nguyen & Zhu, 2013 and Zhen et al., 2009 by using multiple similarity matrices and learning its weights. CABPR method also

significantly differs from MSCMF method, e.g., in the means of content-based similarity incorporation and the learning procedure.

In short, main contributions of this paper are threefold:

- Proposed CABPR method extending original BPR as well as previously published BPR_MCA method.
- Evaluation of CABPR on two publicly available datasets: extended MovieLens 1M[2] and extended LOD-RecSys[3], w.r.t. several cold-start scenarios. CABPR method outperformed BPR_MCA as well as BPR.
- Published extensions to both ML1M and LOD-RecSys datasets available for future work.

1.1 Summary of Notations and Problem Formalization

Before we continue, let us briefly formalize the problem domain and define key notations. For the convenience, notations used outside of their respective defining sections, are also listed in Table 1 in alphabetical order. We denote the set of users as $U = \{u_1, \dots, u_n\}$ and the set of objects as $O = \{o_1, \dots, o_m\}$, where n and m are the number of users and objects respectively. The $n \times m$ matrix **R** represents known user-object interactions (implicit feedback). In the current scenario, **R** is a binary matrix with $r_{i,j} = 1$ denoting that a user u_i positively preferred an object o_j and $r_{i,j} = 0$ otherwise. Function f_S define some,

[2] The original dataset is available from: grouplens.org/datasets/movielens/1m/

[3] The original dataset is available from: challenges.2014.eswc-conferences.org/index.php/RecSys

Table 1: Used notations and descriptions.

Notation	Description
f	volume of latent factors
$f_r;\ f_{\bar{r}}$	latent factor similarity regularization and matrix similarity regularization functions
$f_{\bar{s}}$	cropped objects' or users' similarity function
$m; m_s$	volume of objects; volume of object-based similarity matrices
$n; n_s$	volume of users; volume of user-based similarity matrices
$o_j;\ O;\ O_i^N$	j-th object; vector of all objects; unknown objects for user u_i
$r_{i,j} \in \mathbf{R}$	known preference of user u_i to object o_j
$\hat{r}_{i,j} \in \widehat{\mathbf{R}}$	predicted preference of user u_i to object o_j
$\mathbf{S}_p^U;\ s_{p,i,k}^U$	p-th user-based similarity matrix; similarity of u_i and u_k w.r.t. $\mathbf{S}_p^U$
$\mathbf{S}_q^O;\ s_{q,j,k}^O$	q-th object-based similarity matrix; similarity of o_j and o_k w.r.t. $\mathbf{S}_q^O$
T_s	List of train set examples
$u_i;\ U$	i-th user; vector of all users
η	learning rate
$\Theta;\ \Theta_i$	model parameters; parameters of i-th object or user
$\lambda_{\bar{c}}; \lambda_R; \lambda_w$	content, global and similarity weight regularization hyperparameters
$\mathbf{v}_j; \mathbf{V}$	latent factors of o_j; matrix of all objects' latent factors
$\boldsymbol{\mu}_i;\ \mathbf{U}$	latent factors of u_i; matrix of all users' latent factors
$\sigma(x)$	sigmoid function $\sigma(x) \coloneqq 1/(1+e^{-x})$
$\omega_p; \omega_q; \omega_S; \omega_0$	weight of similarity matrices $\mathbf{S}_p^U$; $\mathbf{S}_q^O$ and $\mathbf{S}$ resp.; uniform similarity weight

possibly latent, CB similarity metric on the pair of users or objects. Matrix induced by such similarity metric is denoted as $\mathbf{S}$. We usually distinguish similarity matrices to user-based and object-based as follows: matrices $\mathbf{S}_p^U \in \mathbb{R}^{(n\times n)}, p \in \{1, \dots, n_s\}$ represent user similarities. Each element $s_{p,i,k}^U$ contains the similarity of users u_i and u_k. Analogically, matrices $\mathbf{S}_q^O \in \mathbb{R}^{(m\times m)}, q \in \{1, \dots, m_s\}$ represent objects similarities. Variables ω_p and ω_q are weights of similarity matrices $\mathbf{S}_p^U$ and $\mathbf{S}_q^O$ respectively.

We further define sets of objects O_i^N, novel for user u_i: $O_i^N = \{o_j \in O;\ r_{i,j} = 0\}$. Matrix factorization methods aim to map both users and objects into a shared latent space, where f denotes its dimension (number of latent factors), $\boldsymbol{\mu}_i \in \mathbb{R}^f$ denotes latent factors of user u_i and $\mathbf{v}_j \in \mathbb{R}^f$ denotes latent factors of object o_j. $\mathbf{U} \in \mathbb{R}^{(n\times f)}$ is the matrix of all users' latent factors and $\mathbf{V} \in \mathbb{R}^{(m\times f)}$ is the matrix of all objects' latent factors. The predicted probability of positive preference $\hat{r}_{i,j}$ for user u_i and object o_j is typically a dot

product of its latent factors $\hat{r}_{i,j} \coloneqq \boldsymbol{\mu}_i \times \boldsymbol{\nu}_j{}^T$. Last, we define the train set of the BPR-based methods as the set of triples $T_s \subset U \times O \times O;\ \ T_s \coloneqq \{(u_i, o_j, o_k) : r_{i,j} = 1 \wedge r_{i,k} = 0\}$, i.e., each train set entry contains user, one of his/her preferred objects and one object, for which we do not have a preference information.

With respect to the introduced notations, the recommending task can be formulated as follows: for a given active user u_i, past preference matrix $\mathbf{R}$ and additional information $\mathbf{S}_p^U, \mathbf{S}_q^O$ predict preferences $\hat{r}_{i,j}$ of objects o_j unknown to the current user ($o_j \in O_i^N$). Top-k objects with the highest $\hat{r}_{i,j}$ values are recommended to the user[4].

2 Materials and Related Work

In order to make this paper self-contained, we would like to first describe three key related concepts fundamental for the proposed method. These are the original BPR method, enhanced gradient-based optimization techniques and existing content-alignment methods incorporating CB information into the matrix factorization.

2.1 Bayesian Personalized Ranking Matrix Factorization

BPR method (Rendle et al., 2009) aims to optimize per-user ranking of objects by reformulating the ranking correctness to the problem of the pairwise ordering correctness for interacting and non-interacting objects. BPR's optimization criterion aims to maximize distances between the predicted ratings $\hat{r}_{i,j} - \hat{r}_{i,k}$ of known preferred object o_j and unknown object o_k for an active user u_i.

---

4 Recommending task may be further refined by applying additional optimization criteria, such as *novelty* or *diversity* (Castells, Hurley, & Vargas, 2015). However, such refinements are typically orthogonal to the recommendation based on estimated relevance of objects and can be combined via post-processing. Therefore, we do not consider such extensions further in this paper.

BPR optimization criterion is formally derived as follows: the Bayesian formulation of finding correct per-user ranking of all objects $o \in O$ is to maximize posterior probability:

$$p(\Theta \mid >_i) \propto p(>_i|\Theta)\, p(\Theta)$$

where Θ represents parameters of the underlined model and $>_i$ is desired, but latent ordering, specific for the user u_i. BPR method further assumes independency of users on each other, independency of ordering pairs of objects on any other pairs, totality and antisymmetry of the ordering. Hence, the user-specific likelihood function $p(>_i |\Theta)$ can be combined for all users as follows:

$$\prod_{u_i \in U} p(>_i|\Theta) = \prod_{(u_i,o_j,o_k) \in T_s} p(o_j >_i o_k|\Theta)$$

The individual probability that user u prefers object o_j over o_k is defined as:

$$p\big(o_j >_i o_k \big|\Theta\big) \coloneqq \sigma(\hat{r}_{i,j,k}(\Theta))$$

where σ is the logistic sigmoid function $\sigma(x) \coloneqq 1/(1+e^{-x})$ and $\hat{r}_{i,j,k}(\Theta)$ is an evaluation function of the underlying model. For matrix factorization, the natural definition of $\hat{r}_{i,j,k}$ is to substract predicted ratings of known and unknown objects: $\hat{r}_{i,j,k} \coloneqq \hat{r}_{i,j} - \hat{r}_{i,k}$. Parameters Θ of matrix factorization are latent factors of users and objects: $\Theta = (\mathbf{U}, \mathbf{V})$. BPR method further assumes prior densities of model parameters to be of normal distribution with zero mean $p(\Theta) \sim N(0, \lambda_\theta I)$, where λ_θ is a model specific regularization parameter. Therefore, the optimization criterion BPR-OPT can be derived as follows (Rendle et al., 2009):

$$
\begin{aligned}
\text{BPR-OPT} &:= \ln p(\Theta \mid >_i) \\
&= \ln p(>_i|\Theta)\, p(\Theta) \\
&= \ln \prod_{(u_i,o_j,o_k)\in T_s} p(o_j >_i o_k|\Theta)p(\Theta) \\
&= \sum_{(u_i,o_j,o_k)\in T_s} \ln\, \sigma\left(\hat{r}_{i,j,k}(\Theta)\right) + \ln\, p(\Theta) \\
&= \sum_{(u_i,o_j,o_k)\in T_s} \ln\, \sigma\left(\hat{r}_{i,j,k}(\Theta)\right) - \lambda_\theta \|\Theta\|^2 \\
&= \sum_{(u_i,o_j,o_k)\in T_s} \ln\, \sigma(\hat{r}_{i,j} - \hat{r}_{i,k}) - \lambda_R(\|\mathbf{U}\|^2 + \|\mathbf{V}\|^2)
\end{aligned}
\tag{1}
$$

The BPR-OPT is optimized via Stochastic Gradient Descend (SGD)[5] over the instances of train set T_s. Train set is usually constructed via bootstrapping unknown objects for the pairs of users and preferred objects. Algorithm 1 shows the pseudocode of the train set construction.

Function construct train set T_s for BPR method
Input: rating matrix **R**, list of unknown objects for each user $[O_i^N]_{i=1}^n$
Output: BPR's train set T_s

```
TS = []
indices = list_nonzero_indices(R)
                          #suppose, indices are a list of tuples [(u_i, o_j)]
foreach (u_i, o_j) in indices:
    o_k = draw_with_uniform_probability(O_i^N)
    TS.append((u_i, o_j, o_k))
return TS
```

Algorithm 1: construction of the BPR's ternary train set

2.2 Gradient-based Iterative Optimization Procedures

Authors of the BPR method proposed to optimize BPR_OPT via Stochastic Gradient Descend (SGD), which is one of the prominent gradient-based optimization

[5] Original BPR-OPT is defined as a maximization criterion, therefore Stochastic Gradient Ascend is used to optimize it. However as both methods are equivalent up to a -1, we continue using better-known SGD term.

methods. However, due to the recent improvements in the area of gradient-based optimization, we aimed to reconsider this choice and therefore this section briefly discuss SGD as well as some other gradient-based optimization methods. First, the (Total) Gradient Descend (GD) method aims to minimize an objective function $f(\Theta)$ by updating parameters Θ in the opposite direction of the gradient of the objective function w.r.t. the learning rate η:

$$\Theta = \Theta - \eta \nabla f(\Theta)$$

However, as calculating the total gradient $\nabla f(\Theta)$ is often time consuming and leads to a slow convergence, SGD instead performs parameter updates based on the gradients of each single training example. More formally, for each function $f_i(\Theta_\mathrm{i})\colon \sum_{\forall i} f_i(\Theta_\mathrm{i}) = f(\Theta)$ and corresponding subset of parameters $\Theta_\mathrm{i} \subset \Theta$, the update step is as follows:

$$\Theta_\mathrm{i} = \Theta_\mathrm{i} - \eta \nabla f_i(\Theta_\mathrm{i})$$

In such way, SGD usually exhibits much faster convergence than GD. Despite its widespread, SGD suffers from some well-known nuisances, such as high variance in the update's gradients, difficult choice of the learning rate, problematical behavior in saddle points and ravines etc. (Ruder, 2017).

A natural way to decrease high variances in updates is to calculate gradients w.r.t. mini-batches of several dozens of training examples instead of a single one. The resulting mini-batch gradient descend method is a reasonable compromise of GD and SGD, offering decreased updates' variance at a reasonable computational cost. Further SGD's improvements were published to deal with the proper learning rate settings and the stability of convergence.

Momentum method (Qian, 1999) combines current and past gradients according to a γ hyperparameter in the update step. In such way, the prevailing gradient direction

starts to dominate the resulting updates and therefore, the proposed method exhibit improved convergence, e.g., in ravines:

$$\delta_{f_i}^{[t+1]} = \gamma\delta_{f_i}^{[t]} + (1-\gamma)\nabla f_i(\Theta_\mathrm{i})$$
$$\Theta_i^{[t+1]} = \Theta_i^{[t]} - \eta\delta_{f_i}^{[t+1]}$$

AdaGrad (Duchi, Hazan, & Singer, 2011) and AdaDelta (Zeiler, 2012) methods aim to estimate proper learning rates for each of the Θ parameters with the underlined assumption that (recently) infrequent parameters should be updated with larger learning rates than the frequent ones. For a small positive ϵ, the update rule is as follows:

$$\Theta_i^{[t+1]} = \Theta_i^{[t]} - \frac{\eta}{\sqrt{g_i^{[t+1]} + \epsilon}}\nabla f_i(\Theta_\mathrm{i})$$

Both approaches differ in the means of g_i calculation. AdaGrad sums squares of all past gradients w.r.t. parameter Θ_i to obtain g_i. Therefore the effective learning rate is non-increasing for all parameters. To mitigate this rather strict behaviour, AdaDelta performs a floating weighted average of squared gradients:

$$g_i^{[t+1]} = \gamma g_i^{[t]} + (1-\gamma)\nabla f_i(\Theta_\mathrm{i})^2$$

ADAM method (Kingma & Ba, 2015) combines AdaDelta for setting parameter-wise learning rate, Momentum to calculate adjusted gradients and per-iteration bias corrections. The final update rule is following (Kingma & Ba, 2015):

$$\begin{aligned}
\delta_{f,i}^{[t+1]} &= \frac{\gamma_1\delta_{f,i}^{[t]} + (1-\gamma_1)\nabla f_i(\Theta_\mathrm{i})}{1-{\gamma_1}^t} \\
g_i^{[t+1]} &= \frac{\gamma_2 g_i^{[t]} + (1-\gamma_2)\nabla f_i(\Theta_\mathrm{i})^2}{1-{\gamma_2}^t} \\
\Theta_i^{[t+1]} &= \Theta_i^{[t]} - \frac{\eta}{\sqrt{g_i^{[t+1]} + \epsilon}}\delta_{f,i}^{[t+1]}
\end{aligned} \qquad (2)$$

Although ADAM as well as other improved gradient-based methods gained a lot of attention in the deep learning domain (Andrychowicz et al., 2016), these were only rarely, if at all, discussed w.r.t. matrix factorization methods for recommender systems.

2.3 Content-based Extensions of Matrix Factorization Models

The idea of incorporating content-based knowledge into the collaborative filtering is not new; see, e.g., Balabanović & Shoham, 1997 for an early approach. Also, several papers proposed to incorporate content-based similarity of users or objects in some matrix factorization technique (e.g., da Silva, Langseth, & Ramampiaro, 2017; Forbes & Zhu, 2011; Krasnoshchok & Lamo, 2014; Lin, Kuo, & Lin, 2014; Nguyen & Zhu, 2013; Zhen et al., 2009). In the related approaches, we can observe two main concepts of content incorporation: *content-driven* and *content-aligned*. Examples of *content-driven* methods are Forbes & Zhu, 2011 and Lin et al., 2014. In these methods, object's latent factors are defined as the sum of its attributes' latent factors:

$$\forall j \in \{1, \dots, m\}: \mathbf{v}_j = \sum_{\forall l} b_{j,l} \times \boldsymbol{\alpha}_l$$

where $\boldsymbol{\alpha}_l$ are latent factors of an attribute a_l and $b_{j,l}$ is a binary information whether object o_j contains attribute a_l. This model optimistically expects that all relevant information, the users process during the evaluation of objects, is expressed in the form of content-based attributes. Such simplification may be quite reasonable on some domains, e.g., recipes recommendation domain, which was evaluated in both papers. In many other domains, however, a lot of information defining desirability of objects may not be easily expressed via CB attributes. For such domains, the *content-aligned* methods may represent a better strategy. The core of content-aligned methods is a similarity metric f_s, assessing the similarity of two users (objects), based on some known CB information. Such similarity functions are both domain-dependent and method-specific. For example,

TagiCoFi (Zhen et al., 2009) used cosine similarity of user-defined tags, variants of CBMF (Nguyen & Zhu, 2013) utilized either the threshold on the dot product of shared attributes, or its sigmoid generalization. The similarity of users (objects) is subsequently applied in some form of additional regularization of similar users' (objects') latent factors, e.g., L2 regularization on the difference of users' latent factors as proposed in TagiCoFi.

$$CA_{TagiCoFi} = \sum_{i=1}^{n} \sum_{\bar{\imath}=1}^{n} f_s(i, \bar{\imath}) \, \|\boldsymbol{\mu}_i - \boldsymbol{\mu}_{\bar{\imath}}\|^2 \tag{3}$$

Such setting prevents similar users (objects) to have too different latent factors. A slightly different regularization was suggested by Nguyen & Zhu:

$$CA_{CBMF} = \sum_{i=1}^{n} \sum_{\bar{\imath}=1}^{n} f_s(i, \bar{\imath}) \, \boldsymbol{\mu}_i{}^T \boldsymbol{\mu}_{\bar{\imath}}$$

However, authors also shown that $CA_{TagiCoFi}$ regularization corresponds with CA_{CBMF} regularization up to the multiplication of $\|\mu_i\|^2$ by a $(1 + 2\sum_{\forall k} f_s(i,k))$ constant. Our preliminary experiments suggested that $CA_{TagiCoFi}$ regularization provides slightly better results than CA_{CBMF} on ML1M dataset.

3 Methods

In this section, we describe the proposed method, *Content-Aligned Bayesian Personalized Ranking* (CABPR) and its variants. Let us first describe, how to utilize simple content alignments in BPR, then extend the model to handle multiple content-based similarities as well as estimating their relevance. After describing the general model, we propose four variants of CABPR, differing in the exact formulas of applied regularizations. Finally, we provide some insight on how to optimize the proposed methods.

3.1 Content-alignments in BPR

We base the content alignments in BPR on the TagiCoFi method (3). An underlined assumption of CABPR (as well as TagiCoFi) is that the independence of users and objects is relaxed by some precomputed similarity function(s) f_s, such that the high similarity of users or objects implies similarity of its respective latent factors. Therefore, differences in latent factors of highly similar users or objects ($\boldsymbol{\mu}_i, \boldsymbol{\mu}_{\bar{\imath}}$, resp. $\boldsymbol{\nu}_j, \boldsymbol{\nu}_{\bar{\jmath}}$) are penalized via some regularization function $f_r(\boldsymbol{\mu}_i, \boldsymbol{\mu}_{\bar{\imath}})$, resp. $f_r(\boldsymbol{\nu}_j, \boldsymbol{\nu}_{\bar{\jmath}})$.

Nonetheless, we would like to stress that such regularization should be imposed only on highly similar users or objects. In real-world applications, there are large volumes of users and objects with only a negligible similarity to each other, approximately following the power law distribution. Regularization w.r.t. these lowly similar pairs could introduce unnecessary noise into the optimization as well as increases algorithm's complexity.

Therefore, we introduce cropped similarities $f_{\bar{s}}$ as follows:

$$f_{\bar{s}}(u_i, u_{\bar{\imath}}) := \begin{cases} f_s(u_i, u_{\bar{\imath}}) \text{ if } f_s(u_i, u_{\bar{\imath}}) \geq f_{s,k,u_i} \\ 0 \text{ otherwise} \end{cases}$$

where f_{s,k,u_i} is the k-th maximal value of $f_s(u_i, \_)$. In order to keep the model simple, we empirically define $k = 10$ in the evaluation. These cropped similarities is one of the key differences between CABPR and MSCMF methods, where the combination of optimization criterion and learning procedure prevents from handling cropped similarities easily.

Now, suppose to have a BPR's train set entry $(u_i, o_j, o_k) \in T_s$, a content regularization hyperparameter λ_c and two similarity metrics $f_{\bar{s}}^U, f_{\bar{s}}^O$ defined on users U and objects O respectively, inducing similarity matrices $\mathbf{S}^U$ and $\mathbf{S}^O$. A simple content-alignment extension of BPR-OPT criterion can be defined as follows:

$$\mathrm{CA}_{BPR_{simple}} = \begin{aligned} &\sum_{i=1}^{n} \sum_{\forall \bar{\imath}:\, s^{U}_{i,\bar{\imath}}>0} \lambda_c \, s^{U}_{i,\bar{\imath}} \; f_r(\mathbf{\mu}_i, \mathbf{\mu}_{\bar{\imath}}) + \\ &\sum_{j=1}^{m} \sum_{\forall \bar{\jmath}:\, s^{O}_{j,\bar{\jmath}}>0} \lambda_c \, s^{O}_{j,\bar{\jmath}} \; f_r\left(\mathbf{\nu}_j, \mathbf{\nu}_{\bar{\jmath}}\right) + \\ &\sum_{k=1}^{m} \sum_{\forall \bar{k}:\, s^{O}_{k,\bar{k}}>0} \lambda_c \, s^{O}_{k,\bar{k}} \; f_r(\mathbf{\nu}_k, \mathbf{\nu}_{\bar{k}}) \end{aligned} \tag{4}$$

Note that it is sufficient to evaluate inner the sums only for elements with non-zero similarities. This suggest to implement $\mathbf{S}^U$ and $\mathbf{S}^O$ as sparse matrices, where such selection is inexpensive.

The next step is to extend CA_{BPR_simple} (4) to handle arbitrarily many similarity matrices $\mathbf{S}^U_p$, $\mathbf{S}^O_q$ and to learn relevance of each similarity matrix. In order to do so, we decompose the content regularization hyperparameter λ_c into a fixed part $\lambda_{\bar{c}}$ and variable parts ω_S specific for each similarity matrix $\mathbf{S}$. Naturally, to prevent overfitting, some regularization $f_{\bar{r}}$ should be applied also on ω_S values with a regularization weight hyperparameter λ_w. With respect to these assumptions, the content alignments extension of BPR is as follows:

$$CA_{BPR} = \begin{aligned} &\textstyle\sum_{p=1}^{n_s} \sum_{i=1}^{n} \sum_{\forall \bar{\imath}:\, s^{U}_{i,\bar{\imath}}>0} \lambda_{\bar{c}} \, \omega_p \, s^{U}_{i,\bar{\imath}} \; f_r(\mathbf{\mu}_i, \mathbf{\mu}_{\bar{\imath}}) + \\ &\textstyle\sum_{q=1}^{m_s} \sum_{j=1}^{m} \sum_{\forall \bar{\jmath}:\, s^{O}_{j,\bar{\jmath}}>0} \lambda_{\bar{c}} \, \omega_q \, s^{O}_{j,\bar{\jmath}} \; f_r\left(\mathbf{\nu}_j, \mathbf{\nu}_{\bar{\jmath}}\right) + \\ &\textstyle\sum_{q=1}^{m_s} \sum_{k=1}^{m} \sum_{\forall \bar{k}:\, s^{O}_{k,\bar{k}}>0} \lambda_{\bar{c}} \, \omega_q \, s^{O}_{k,\bar{k}} \; f_r(\mathbf{\nu}_k, \mathbf{\nu}_{\bar{k}}) + \\ &\lambda_w \textstyle\sum_{S \in \{\mathbf{S}^U_p, \mathbf{S}^O_q\}} f_{\bar{r}}\,(\omega_s) \end{aligned} \tag{5}$$

The final CABPR-OPT maximization criterion is derived by summing the content alignments (5) together with other regularizations of BPR-OPT (1):

$$\mathrm{CABPR\_OPT} = \sum_{(u_i, o_j, o_k) \in T_s} \ln \, \sigma\left(\hat{r}_{i,j} - \hat{r}_{i,k}\right) - \lambda_R(\|\mathbf{U}\|^2 + \|\mathbf{V}\|^2 + \mathrm{CA}_{BPR}) \tag{6}$$

3.2 Variants of CABPR

Note that in the description of CABPR, we did not yet define the regularization functions f_r and $f_{\bar{r}}$. In this paper, we propose two variants for both f_r and $f_{\bar{r}}$. As the choices are orthogonal, there are in total four variants of CABPR method. Please note that in the following definitions, we show only the user-based formulas of f_r regularizations. The object-based definitions are analogical.

As a first variant of f_r, we utilized the TagiCoFi approach, i.e., L2 regularization on the differences of latent factors:

$$L^2(\boldsymbol{\mu}_i, \boldsymbol{\mu}_{\bar{\imath}}) \coloneqq \|\boldsymbol{\mu}_i - \boldsymbol{\mu}_{\bar{\imath}}\|^2 \tag{7}$$

The choice of L^2 regularization corresponds with the intended prior probability $p(\boldsymbol{\mu}_i)$ to be of a multivariate normal distribution with the means being a list of $\boldsymbol{\mu}_{\bar{\imath}}$ for all $\bar{\imath}$ and zero (corresponding with the original BPR's regularization).

With the second choice of f_r, we aimed on relaxing the required latent factor's similarity. The intuition behind is that although we want similar users (objects) to have similar latent factors, they do not necessarily have to be too close as CB similarity is merely a supporting evidence to the collaborative knowledge. Therefore, we may specify an allowed margin ε and penalize only differences beyond this margin:

$$L^2_{WM}(\boldsymbol{\mu}_i, \boldsymbol{\mu}_{\bar{\imath}}) \coloneqq \max(0, \|\boldsymbol{\mu}_i - \boldsymbol{\mu}_{\bar{\imath}}\|^2 - \varepsilon) \tag{8}$$

The definition of L^2_{WM} is based on the L_D term in contrastive loss function (Hadsell, Chopra, & LeCun, 2006)[6].

The variants of $f_{\bar{r}}$ are derived as follows. First, for the sake of consistency, we wish to maintain the overall weight of the content-alignments in CA_{BPR} to be proportional

---

[6] To be more specific, we use the opposite of the second maximization term in $L_D \coloneqq \frac{1}{2}\max(0, \varepsilon - \|\boldsymbol{\mu}_i - \boldsymbol{\mu}_{\bar{\imath}}\|^2)^2$

to λ_c, i.e., $\lambda_{\bar{c}} \sum_{\forall S} |\omega_S| \approx \lambda_c$. Furthermore, to prevent overfitting, we prefer simpler weighting models not too different from the uniform distribution of weights. By combining those two principles, we aim to penalize the difference of ω_S and $\omega_0 \coloneqq \frac{1}{n_s+m_s}\lambda_c$.

In the early stages of our work, we experimented with a plain squared regularization $L^2(\omega_S) \coloneqq (\omega_S - \omega_0)^2$, corresponding with the normal prior distributions of ω_S values around the mean at ω_0.

However, L^2 regularization allows ω_S to reach negative values, implying possibly positive CA_{BPR} (5) regularization term in the maximization procedure. Eventually, the CA_{BPR} term may dominate the whole CABPR-OPT criterion (6) and diverge the optimization. Therefore, different variants of $f_{\bar{r}}$ were proposed and evaluated to cope with this problem.

First, the *fixed* total *content weight* $(L^2 + FCW)$ regularization preserve the squared regularization as is and defines an additional strict requirement that the sum of absolute content weights equals to λ_c:

$$\sum_{\forall S \in \{\mathbf{s}_p^U, \mathbf{s}_q^O\}} \lambda_{\bar{c}} |\omega_S| = \lambda_c \tag{9}$$

The equality is checked after each optimization step and content weights are linearly scaled to adhere with the condition. This adjustment does not prevent the model from divergence, however, by bounding the minimal weights of ω_S, it reduces its probability. On the other hand, the possibility to learn negative content weights is retained, allowing the model to detect unexpected content-based dissimilarities.

In the second variant of $f_{\bar{r}}$, we bound ω_S values to be from $(0, 1)$ interval by gradually increased penalty by the factor of $\frac{1}{\omega_S}$ and $\frac{1}{1-\omega_S}$ respectively.

$$L^1_{NN}(\omega_S) \coloneqq |\omega_S - \omega_0| \left(\frac{1}{\omega_S} + \frac{1}{1-\omega_S}\right) \quad (10)$$

In the evaluation, the variants of CABPR are distinguished according to the selected f_r and $f_{\bar{r}}$ variants, e.g., CABPR-L2-NN denotes the CABPR method, where $f_r = L^2$ and $f_{\bar{r}} = L^1_{NN}$.

3.3 Optimization Procedure

Based on the literature review, we decided to change the optimization procedure to ADAM optimization method with mini-batch processing. In the evaluation, we kept the default ADAM parameters, i.e., $\gamma_1 = 0.9$, $\gamma_2 = 0.999$ and $\epsilon = 10^{-8}$ and use the mini-batches of the size 128. Algorithm 2 provides a pseudo-code for the final CABPR training procedure.

Function train CABPR method
Input: f, η, λ_R, $\lambda_{\bar{C}}$, λ_w, ω_0, *max_iterations*, *batch_size*, *CABPR-OPT*
Output: users' and objects' latent factors **U** and **V**

```
1: Initialize U and V at random
2: Initialize ω_S = ω_0
3: Instantiate CABPR-OPT with f, λ_R, λ_C̄, λ_w, f_r and f_r̄ params
4: Repeat until max_iterations is reached:
5:   T_S = construct_train_set() #Algorithm 1
6:   Repeat until b = next_batch(T_S, batch_size) exists:
7:     Update U, V, ω_S via ADAM(CABPR-OPT, b, η) #Equation (2)
8:     If f_r̄ == (L^2 + FCW): scale ω_S so that (9) holds.
9: return U and V
```

Algorithm 2: training CABPR method. Note that *ADAM()* procedure on line 7 corrersponds with a call of TensorFlow *AdamOptimizer.maximize()* method. *Next_batch()* procedure on line 6 returns the next *batch_size* items from the train set T_S or *False* if there are no more available items.

4 Datasets

We evaluated CABPR on two publicly available datasets: MovieLens 1M dataset, which was already used in the previous work and LOD-RecSys dataset. Both datasets were

extended to contain relevant content-based information.

4.1 Extended MovieLens 1M Dataset

The first dataset is based on a well-known MovieLens 1M (ML1M) dataset (Harper & Konstan, 2015). The original dataset contains over one million ratings on 1 to 5 scale from 6041 users and in total 3883 movies. Furthermore, the dataset contains some basic user statistics, such as gender, occupation, age group and a ZIP code. Title, year of publication and genres are available for the movies. As the input of BPR method is a binary user rating, the original graded feedback was binarized as follows: all known interactions with user rating ≥ 3 were considered as positive, i.e., $r_{u,o} = 1$, all other interactions (unrated and those with rating < 3) were considered as unknown, i.e., $r_{u,o} = 0$. ML1M dataset was extended by content-based information from three sources:

- User profiles were extended via ZIP code statistics.
- Item profiles were extended via IMDB movie features.
- Item profiles were extended via DBTropes features.

Figure 1 depicts the general schema of the dataset extensions. Based on the provided US ZIP codes, we collected data from *UnitedStatesZipCodes.org* website, to describe the neighborhood of the user. Collected features aims to distinguish between urban and countryside areas (*total population*, *density*), social picture (*median age*, *singles ratio*), economic situation (*vacant houses ratio*) and ethnicity of the inhabitants. In total, we were able to collect ZIP code statistics about 5875 out of 6041 users. Note that although ZIP codes are not commonly provided by the users, similar information can be obtained based on their IP addresses.

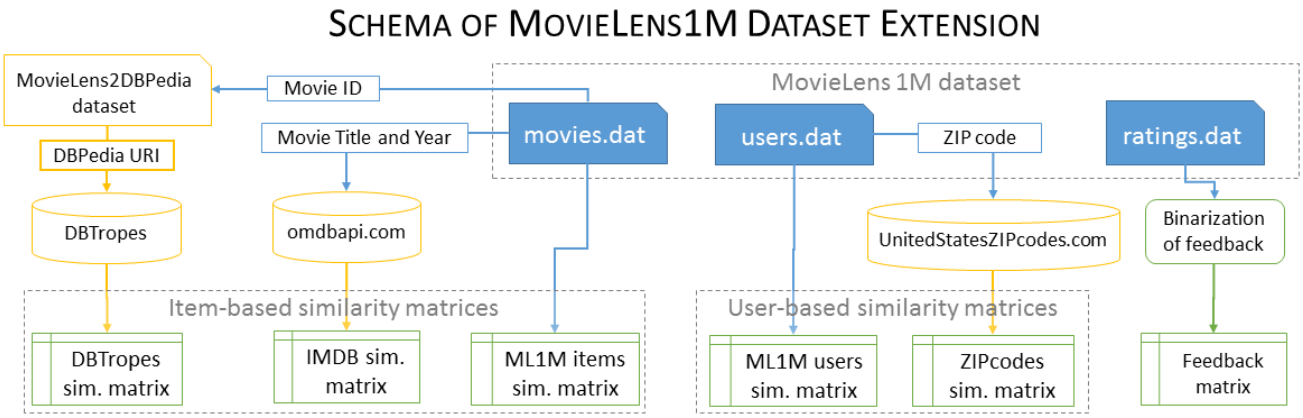

Figure 1: Schema of the extensions made on the MovieLens 1M dataset.

In order to define finer grained similarity on movies, we queried IMDB API[7] with the query based on the movie title and the year of production. In total, we were able to match 3085 movies of the original ML1M dataset. Collected features foremost link movies through the contributing persons (*actors*, *director*, *writer*) and provide further general description (*language*, *country of origin*, *average rating*).

Although the IMDB metadata already provide relevant features of the movies, these features are mostly "external" (i.e., providing contributing persons and circumstances rather than describing relevant plot features). In order to bridge this semantic gap, we also collected plot characteristics available from DBTropes (Kiesel & Grimnes, 2010), a linked open data extension of TvTropes[8]. TvTropes community focus on identifying and describing behavior archetypes, typical story lines and common character types that occurs across the movies. As such, TvTropes represents ideal source of information for an "intrinsic" movie similarity metric. DBTropes were linked to ML1M dataset via shared DBPedia identifier. The coverage of this dataset was, however, rather low; we were able to match 589 movies of the original ML1M dataset.

[7] http://www.omdbapi.com

[8] http://tvtropes.org/

For each of the content-based sources, we define a separate similarity metric as the *cosine similarity* of binarized attributes' TF-IDF score. The resulted extended ML1M dataset contains five similarity matrices, two user-based and three object-based. More details on extended ML1M dataset can be found in (Ladislav Peska, 2017).

4.2 Extended LOD-RecSys Dataset

The second dataset is based on the ESWC 2014 Linked Open Data enabled Recommender Systems Challenge (LOD-RecSys; Di Noia, Cantador, & Ostuni, 2014). The challenge focused on the book domain with the data collected from the LibraryThing[9] website. We utilized the train dataset of the Task 1, containing 75559 ratings on the 0-5 scale, 6181 users and 6166 items. For each object, the dataset provides its DBPedia URI, but no other content-based information are available. Similarly as in extended ML1M dataset, user ratings were binarized such that all known interactions with rating ≥ 3 were considered as positive, i.e., $r_{u,o} = 1$, otherwise $r_{u,o} = 0$. As we do not have any information about users, all extensions were based on the objects' DBPedia URI. The dataset was extended by five types of content-based information collected via DBPedia[10].

- Explicit books similarity.
- Books of the same author.
- Similarity of books based on shared direct categories.
- Similarity of books based on shared extended categories.
- Similarity of books based on merged authors' and books' properties.

[9] https://www.librarything.com/

[10] http://dbpedia.org

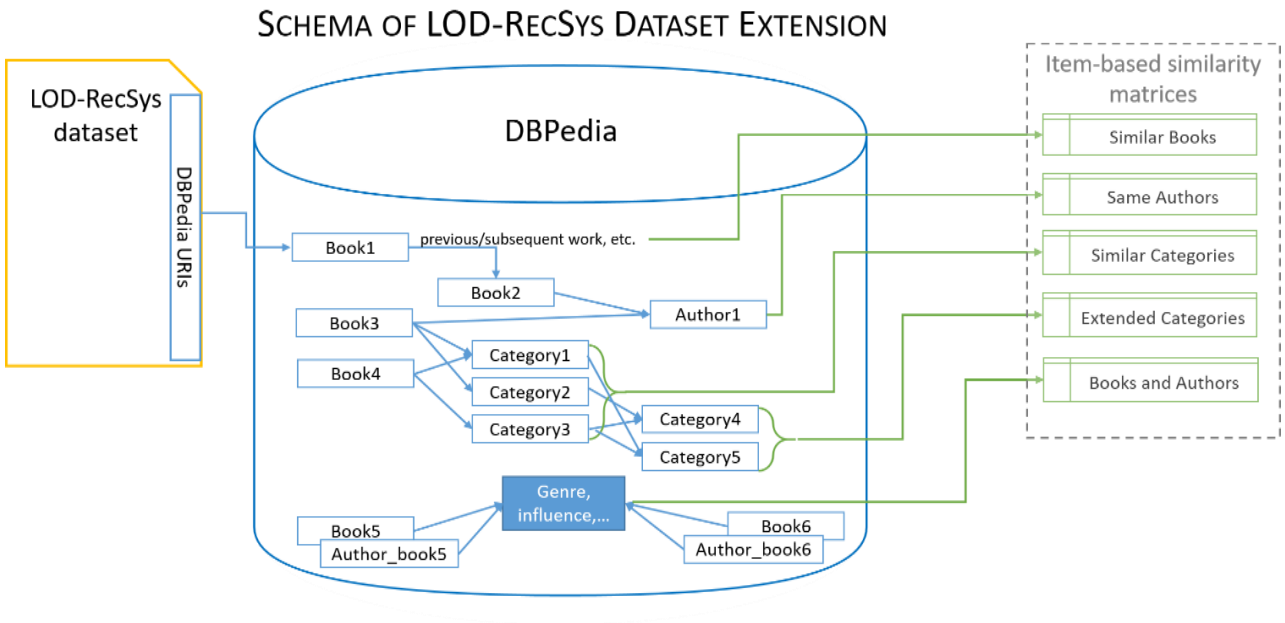

Figure 2: Schema of the extensions made on the LOD-RecSys dataset.

Figure 2 depicts the schema of extensions made on the LOD-RecSys dataset. While considering possible extensions of the LOD-RecSys dataset, we followed several commonly applied notions of similarity specific for the books domain. First, we aimed to identify pairs of books, which were explicitly denoted as similar in DBPedia. Those are, e.g., the books from the same book series. The notion of similarity is spread across several properties in DBPedia, therefore we consider a book o_j to be explicitly similar with book o_i if there is an RDF triple (o_i, p, o_j), where p is a property, which explicitly defines similarity, e.g., *dbo:basedOn*, *dbo:previousWork*, *dbo:subsequentWork*, *dbp:before*, *dbp:after* etc.

Another commonly used notion of similarity is whether the books were written by the same author. Although an author may divide his/her creative work into several domains, which may also change over time, the uniqueness of ideas, writing style or narrative still provides a strong distinctive power. Therefore, we consider books o_i, o_j as similar, if they share an author a, i.e., the following subgraph exists: $(o_i, \text{dbo:author}, a)$, $(o_j, \text{dbo:author}, a)$. In both explicit similarity and books of the same author, the resulting

similarity matrices $\mathbf{S}_k^O$ are of binary values, i.e., $s_{k,i,j}^O = 1$ if the books o_i and o_j possess the considered feature and 0 otherwise.

Both of the previously mentioned extensions provide rather straightforward notions of similarity, which are well backed in real-life usage, but are rather too narrow and unsurprising on the other hand. Therefore, we also focused on a bit more speculative notions of similarity, which can, however, provide more surprising, yet relevant results.

First, we aimed to utilize the effort of Wikipedia contributors on categorization of information. The vast majority of Wikipedia articles are classified into several highly specific hierarchical categories, e.g., "*Novels about death*", "*Discworld books*", "*1987 British novels*" and "*1980s fantasy novels*" for the Mort novel by Terry Pratchet. In DBPedia, these categories are available through *dct:subject* property, while the more general categories can be accessed through *skos:broader* property of each category. We defined two similarity matrices based on the categories: direct and extended. The similarity based on directly shared categories utilized the *dct:subject* property, i.e., for each book o_i, we collect the list of categories C_i such that $\forall c \in C_i: (o_i, \mathrm{dct:subject}, c)$ triple exists. The similarity of books o_i, o_j is defined as a Jaccard similarity of their categories C_i, C_j. The similarity on extended categories starts with the set of direct categories C and extends it by super-categories c_s, such that $\exists c_i \in C: (c_i, \mathrm{skos:broader}, c_s)$. We considered two levels of super-categories and the resulting similarity is a Jaccard similarity of extended category sets.

Finally, the last similarity matrix aims to maximize the coverage among books. As DBPedia often contains much richer information about authors than some of their books, we chose to incorporate also relevant authors' properties to the similarity calculation. Therefore, we are able to link similar books through the similarity of authors even in cases, where the lack of information about books prevents direct reasoning. The

similarity is defined as a Jaccard similarity of merged relevant books' and authors' features, e.g., *dbo:literaryGenre*, *dbp:genre*, *foaf:primaryTopic*, *dbo:influences*, *dbo:influencedBy*.

5 Evaluation and Results

In this section, we would like to provide details of the evaluation procedure. It was shown that collaborative techniques need the support of content-based or non-personalized recommendations especially in cold-start scenarios (Kluver & Konstan, 2014; Pilászy & Tikk, 2009), which are however quite common in the real-world applications (Kaminskas, Bridge, Foping, & Roche, 2017; L. Peska & Vojtas, 2014, 2017). Therefore, the evaluation focused on the performance of proposed method w.r.t. several, increasingly difficult cold-start scenarios. Table 2 provides overview of the evaluated methods.

Hyperparameters of evaluated methods were learned via grid-search and internal bootstrap sampling as follows. The number of latent factors was fixed $f = 20$, initial learning rate was fixed $\eta = 0.1$, regularization λ_R was selected from $\{0.01, 0.05\}$ and the number of iterations was selected from $\{5, 10, 15, \dots, 30\}$. Furthermore, for CABPR, the content regularization $\lambda_{\bar{C}}$ was selected from $\{0.001, 0.01, 0.05\}$ and similarity weight regularization λ_w from $\{0.01, 0.05, 0.2\}$. The ε hyperparameter of L^2_{WM} was kept constant at 0.1.

Table 2: Overview of evaluated methods.

Method	f_r	$f_{\bar{r}}$	**Optimization**
BPR (Rendle et al., 2009)	-	-	SGD
BPR_MCA (Ladislav Peska, 2017)	L^2 (7)	$L^2 + FCW$ (9)	SGD
CABPR-L2-FCW	L^2 (7)	$L^2 + FCW$ (9)	Adam + mini-batches
CABPR-L2-NN	L^2 (7)	L^1_{NN} (10)	Adam + mini-batches
CABPR-WM-FCW	L^2_{WM} (8)	$L^2 + FCW$ (9)	Adam + mini-batches
CABPR-WM-NN	L^2_{WM} (8)	L^1_{NN} (10)	Adam + mini-batches

Due to the need of cold-start simulation, we opted for a Monte Carlo cross-validation with low train/test ratios. As ML1M dataset is much denser than LOD-RecSys, we used different thresholds, which roughly maintains the same sparsity levels.

For ML1M, three scenarios were evaluated, with 90%, 95% and 98% of the interactions randomly removed from the interaction matrix **R** for training. These scenarios are denoted as *p90*, *p95* and *p98* in the results. Also for LOD-RecSys dataset, three scenarios were evaluated with 50%, 75% and 90% of the interactions removed. Each training scenario was repeated 5-times, while the hidden interactions were used for per-user evaluation. We used normalized discounted cumulative gain (nDCG) and area under precision-recall curve (AUPR) as evaluation metrics. NDCG logarithmically penalizes relevant objects appeared too low in the recommended list. AUPR is considered to reflect algorithm's performance well in case of highly imbalanced data (Davis & Goadrich, 2006), so both metrics seems to be a reasonable approximation of top-k ranking evaluation problem. Both metrics were evaluated for each user and averaged results over all users and all CV runs are provided.

5.1 Results and Discussion

The overall results on ML1M dataset are depicted on Table 3 and the results on LOD-RecSys dataset are depicted on Table 4. As can be seen from the results, CABPR methods significantly outperform both original BPR and BPR_MCA methods w.r.t. two more difficult cold-start scenarios in both ML1M and LOD-RecSys datasets.

Table 3: Overall results of the evaluation on extended ML1M dataset. Best results are in bold, baseline methods are in grey, italic. Results with significantly lower performance than the best method ($p<0.05$ according to the t-test) are marked with an asterisk (*). Note that due to the divergence of its optimization, we do not provide CABPR-WM-FCW results

	nDCG			**AUPR**		
Method	***p90***	***p95***	***p98***	***p90***	***p95***	***p98***
BPR	**0.6351*	**0.5739*	**0.5490*	**0.1976*	**0.1147*	**0.0895*
BPR_MCA	***0.6425***	**0.6016*	**0.5646*	***0.2045***	**0.1485*	**0.1030*
CABPR-L2-FCW	*0.6310	**0.6338**	**0.6216**	*0.1794	0.1757	**0.1636**
CABPR-L2-NN	*0.6228	0.6319	0.6196	*0.1732	**0.1762**	0.1628
CABPR-WM-FCW	N/A	N/A	N/A	N/A	N/A	N/A
CABPR-WM-NN	*0.6262	*0.6270	0.6202	*0.1760	*0.1734	0.1634

As for the best performing variant of CABPR, there is no single optimal method w.r.t. Pareto front. CABPR-WM-FCW method in general provided inferior results and its optimization often diverged, therefore, we would not recommend this variant. As for the other three variants, CABPR with non-negative content weights (NN) performed better on LOD-RecSys dataset, while CABPR-L2-FCW achieved slightly better results on ML1M dataset.

However, the differences between CABPR variants were often not significant, or negligible if compared to the performance of other algorithms. Therefore, as an auxiliary criterion, we also evaluated temporal complexity of the proposed methods. Table 5

Table 4: Overall results of the evaluation on extended LOD-RecSys dataset. Best results are in bold, baseline methods are in grey, italic. Results with significantly lower performance than the best method ($p<0.05$ according to the t-test) are marked with an asterisk (*).

	nDCG			**AUPR**		
Method	***p50***	***p75***	***p90***	***p50***	***p75***	***p90***
BPR	***0.2953***	**0.2682*	**0.2562*	***0.0543***	**0.0287*	**0.0199*
BPR_MCA	**0.2932*	**0.2668*	**0.2572*	**0.0524*	**0.0278*	**0.0195*
CABPR-L2-FCW	*0.2822	*0.2987	0.3069	*0.0419	*0.0382	0.0380
CABPR-L2-NN	*0.2829	**0.3005**	0.3068	*0.0426	**0.0402**	0.0380
CABPR-WM-FCW	*0.2853	*0.2960	0.3066	*0.0440	*0.0361	0.0377
CABPR-WM-NN	*0.2820	0.2998	**0.3073**	*0.0424	0.0399	**0.0383**

Table 5: Temporal complexity of the variants of CABPR algorithm (average time per iteration in seconds on LOD-RecSys dataset). CABPR was implemented in Python using TensorFlow, evaluation was performed on Intel Core i7 PC with 16GB RAM and NVIDIA GeForce GTX 1050 GPU.

	Average time per iteration (sec)		
Method	***p50***	***p75***	***p90***
CABPR-L2-FCW	373.03	104.01	29.11
CABPR-L2-NN	**71.45**	35.14	**13.78**
CABPR-WM-FCW	274.68	129.97	30.29
CABPR-WM-NN	71.60	**34.85**	14.28

contains the average time per iteration for CABPR variants. It can be seen, that CABPR with fixed content weights (FCW) is considerably more expensive to train and therefore, we may recommend usage of CABPR-L2-NN and CABPR-WM-NN methods.

5.1.1 The role of batch size

The *batch size* hyperparameter defines the number of train set items optimized at the same time. Increasing *batch size* leads to a decreased variance in parameter updates, but may also cause the decrease of method's flexibility and higher risk of convergence to a local optimum. One of the surprising result of the evaluation was the decrease of CABPR's performance for increasing train set sizes. The effect was more visible for nDCG and LOD-RecSys dataset (Table 4). This observation may be partially attributed to the decreased frequency of positive examples in test sets as well as decreased importance of content-based over collaborative information (Kluver & Konstan, 2014; Pilászy & Tikk, 2009). However, as the performance of BPR_MCA method increased between the respective scenarios, we decided to evaluate, whether a portion of observed behavior may be also attributed to the lack flexibility caused by too high *batch size* settings.

Table 6: Results of CABPR methods for various settings of *batch size* hyperparameter. Best results are in bold, baseline methods are in grey, italic. Results with significantly lower performance than the best method ($p<0.05$ according to the t-test) are marked with an asterisk (*).

ML1M dataset	**Batch size**	***nDCG***			***AUPR***		
		p90	***p95***	***p98***	***p90***	***p95***	***p98***
Best@Table 3	-	*0.6425	*0.6338	*0.6216	0.2045	*0.1762	*0.1636
CABPR-L2-NN	*128*	*0.6228	*0.6319	*0.6196	*0.1732	*0.1762	*0.1628
CABPR-L2-NN	32	**0.6520**	**0.6493**	**0.6462**	**0.2058**	**0.2009**	**0.1933**
CABPR-L2-NN	8	*0.6383	*0.6425	*0.6357	*0.1899	*0.1910	*0.1806
LOD-RecSys		***p50***	***p75***	***p90***	***p50***	***p75***	***p90***
Best@Table 4	-	*0.2953	*0.3005	*0.3073	*0.0543	*0.0402	*0.0383
CABPR-L2-NN	*128*	*0.2829	*0.3005	*0.3068	*0.0426	*0.0402	*0.0380
CABPR-L2-NN	32	**0.3029**	**0.3151**	**0.3193**	**0.0560**	**0.0477**	**0.0445**
CABPR-L2-NN	8	*0.2979	*0.3106	*0.3154	*0.0526	*0.0458	*0.0427

Table 6 depicts the results of evaluation w.r.t. *batch size*. It can be seen that methods with lower *batch sizes* provided better results than the original settings. Furthermore, *CABPR-L2-NN* method with batch size of 32 outperformed all baseline methods w.r.t. both nDCG and AUPR and all cold start settings. Nonetheless, although a decreased performance for increased train set size was not observed for the best method in ML1M dataset any more, it persisted in nDCG evaluation of LOD-RecSys dataset and remained rather consistent for all evaluated batch sizes. We hypothesize that in this case, the main cause is the decreased frequency of positively preferred objects in the test set, which is a natural consequence of selected evaluation methodology. The absence of such behaviour in BPR_MCA results could be attributed to relatively better adaptation of CABPR to lower amount of train data due to the improved optimization procedure.

A bit more speculative hypothesis is that CABPR-OPT criterion largely resembles AUC metric, which differs from evaluated nDCG and the better convergence towards AUC (supported with more evidence while increasing the train size) may eventually result in worse nDCG performance. This is illustrated by the results w.r.t. AUC (available in supplementary materials), which is non-decreasing also for LOD-RecSys dataset.

Therefore, as a part of our future work, we plan to evaluate alternative optimization criteria that better correspond with graded relevance metrics, such as nDCG.

5.1.2 Weights of similarity matrices

As the ability to learn importance of similarity matrices is one of the key CABPR's features, we also evaluated, whether the learned similarity weights differ from one another. Figure 3 and Figure 4 depict boxplots of the final similarity matrices' weights ω_S learned during the evaluation of ML1M and LOD-RecSys datasets respectively.

It can be seen that although CABPR with fixed total content weight (FCW) allows weights of similarity matrices to grow negative, such weights were not learned in the final models (with a single exception). During the hyperparameter tuning phase, negative values occurred occasionally, however mostly provided inferior results. Therefore, we may hypothesize that the negative weights of some similarity matrices learned by BPR_MCA method during the previous experiments (Ladislav Peska, 2017) may be attributed to the problem of high variances in SGD updates rather than irrelevance of certain similarity matrices.

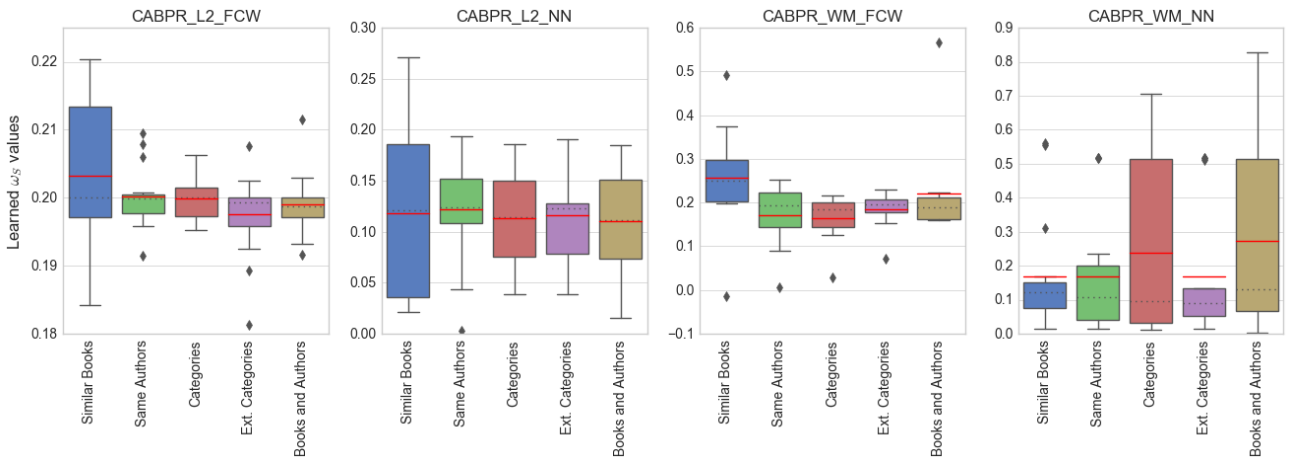

Figure 4: Boxplots of the final content weights ω_S learned for LOD-RecSys dataset, aggregated for all cross-validation runs and cold-start settings. Red full line denotes mean and black dotted line median of the values.

Despite the fact that some CABPR variants seemed to learn weights close to the uniform distribution (e.g., CABPR_WM_NN in ML1M dataset), the overall difference in the learned weights was rather high and therefore we may conclude that the estimation of content weights is an important aspect of the learning algorithm. We can also corroborate the importance of intrinsic features received from *DBTropes* dataset, for which, high weights were learned in most cases. As for the LOD-RecSys dataset, high weights were often learned for explicitly *similar books* and in case of CABPR_WM_NN method also

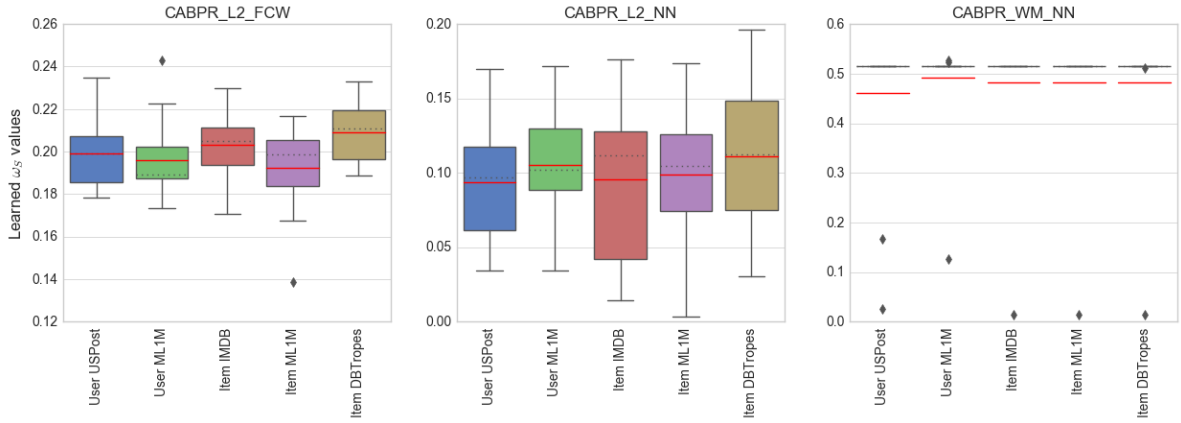

Figure 3: Boxplots of the final content weights ω_S learned for ML1M dataset, aggregated for all cross-validation runs and cold-start settings. Red full line denotes mean and black dotted line median of the values.

for books with similar *categories* and merged *books and authors* similarity. In general, CABPR variants with L^2_{WM} latent factor regularization learned considerably higher similarity weights than the ones with strict L^2 regularization, corroborating the effect of similarity relaxation parameter ε.

6 Conclusions

In this paper, we proposed content-alignments extension to the BPR matrix factorization, CABPR. CABPR incorporates content alignments based on multiple CB similarity matrices as additional regularizations into the BPR's optimization criterion. Four variants of the CABPR method were proposed, differing in the exact regularization formula.

An extended MovieLens 1M dataset as well as an extended LOD-RecSys dataset were used to evaluate the proposed methods. All CABPR variants provided significant improvements over standard BPR as well as previously published BPR_MCA in several difficult cold-start settings w.r.t nDCG and AUPR. Based on the overall performance, training time and stability, we recommend CABPR with *non-negative* content weights as the best choice. Additional experiments shown the importance of proper batch size settings, where the optimal values were slightly lower (best results were reported for the batch size of 32) than the ones originally evaluated. Evaluation also shown a significant difference in learned weights of similarity matrices, corroborating the importance of evaluating relevance of content sources.

There are several directions, which may be addressed in the future work. As the CABPR is mostly orthogonal to various diversity or novelty enhancing approaches, these can be integrated while transferring CABPR to a production environment. Similarly, CABPR can be extended to handle graded or implicit negative feedback, if such data is available.

The importance of using intrinsic features of objects was already pointed out in our previous work and corroborated in current experiments. In domains with important multimedia content (such as movies, music, fashion etc.), CABPR may utilize the similarity based, e.g., on the output feature vectors of recurrent or convolutional neural networks and therefore bridge the semantic gap of external vs. intrinsic content-based features. A direct future work in this direction would be to integrate CABPR to the fashion product exploration and recommendation tool recently published by our group (Skopal, Peška, Kovalcík, Grošup, & Lokoc, 2017).

Acknowledgements: CABPR was implemented in Python using TensorFlow framework. Source codes can be obtained from *github.com/lpeska/CABPR*. Datasets and supplementary materials can be downloaded from *www.ksi.mff.cuni.cz/~peska/CABPR/*. The work on this paper was supported by Czech grants Q48 and GACR-17-22224S.

References

Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M. W., Pfau, D., Schaul, T., & de Freitas, N. (2016). Learning to learn by gradient descent by gradient descent. *NIPS*, 1–16. https://doi.org/10.1007/s10115-008-0151-5

Balabanović, M., & Shoham, Y. (1997). Fab: content-based, collaborative recommendation. *Communications of the ACM*, *40*(3), 66–72. https://doi.org/10.1145/245108.245124

Castells, P., Hurley, N. J., & Vargas, S. (2015). Novelty and diversity in recommender systems. In *Recommender Systems Handbook, Second Edition* (pp. 881–918). https://doi.org/10.1007/978-1-4899-7637-6_26

da Silva, E. de S., Langseth, H., & Ramampiaro, H. (2017). Content-Based Social Recommendation with Poisson Matrix Factorization. In M. Ceci, J. Hollmén, L. Todorovski, C. Vens, & S. Džeroski (Eds.), *Machine Learning and Knowledge Discovery in Databases* (pp. 530–546). Cham: Springer International Publishing.

Davis, J., & Goadrich, M. (2006). The relationship between Precision-Recall and ROC curves. In *Proceedings of the 23rd international conference on Machine learning - ICML '06* (pp. 233–240). https://doi.org/10.1145/1143844.1143874

de Campos, L. M., Fernández-Luna, J. M., Huete, J. F., & Rueda-Morales, M. A. (2010). Combining content-based and collaborative recommendations: A hybrid approach based on Bayesian networks. *International Journal of Approximate Reasoning*, *51*(7), 785–799. https://doi.org/http://dx.doi.org/10.1016/j.ijar.2010.04.001

Di Noia, T., Cantador, I., & Ostuni, V. C. (2014). Linked Open Data-enabled Recommender Systems: ESWC 2014 Challenge on book recommendation. *Communications in Computer and Information Science*, *475*, 129–143. https://doi.org/10.1007/978-3-319-12024-9_17

Duchi, J., Hazan, E., & Singer, Y. (2011). Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, *12*, 2121–2159. https://doi.org/10.1109/CDC.2012.6426698

Forbes, P., & Zhu, M. (2011). Content-boosted matrix factorization for recommender systems: experiments with recipe recommendation. In *Proceedings of the fifth ACM conference on Recommender systems* (pp. 261–264). New York, NY, USA: ACM. https://doi.org/10.1145/2043932.2043979

Hadsell, R., Chopra, S., & LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (Vol. 2, pp. 1735–1742). https://doi.org/10.1109/CVPR.2006.100

Harper, F. M., & Konstan, J. A. (2015). The MovieLens Datasets. *ACM Transactions on Interactive Intelligent Systems*, *5*(4), 1–19. https://doi.org/10.1145/2827872

Hu, Y., Koren, Y., & Volinsky, C. (2008). Collaborative Filtering for Implicit Feedback

Datasets. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining* (pp. 263–272). Washington, DC, USA: IEEE Computer Society. https://doi.org/10.1109/ICDM.2008.22

Johnson, C. C. (2014). Logistic Matrix Factorization for Implicit Feedback Data. *NIPS 2014 Workshop on Distributed Machine Learning and Matrix Computations*, 1–9.

Kaminskas, M., Bridge, D., Foping, F., & Roche, D. (2017). Product-Seeded and Basket-Seeded Recommendations for Small-Scale Retailers. *Journal on Data Semantics*, *6*(1), 3–14. https://doi.org/10.1007/s13740-016-0058-3

Kiesel, M., & Grimnes, G. A. (2010). DBTropes-A linked data wrapper approach incorporating community feedback. In *CEUR Workshop Proceedings* (Vol. 674).

Kingma, D. P., & Ba, J. L. (2015). Adam: a Method for Stochastic Optimization. *International Conference on Learning Representations 2015*, 1–15. https://doi.org/http://doi.acm.org.ezproxy.lib.ucf.edu/10.1145/1830483.1830503

Kluver, D., & Konstan, J. A. (2014). Evaluating Recommender Behavior for New Users. In *Proceedings of the 8th ACM Conference on Recommender Systems* (pp. 121–128). New York, NY, USA: ACM. https://doi.org/10.1145/2645710.2645742

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix Factorization Techniques for Recommender Systems. *Computer*, *42*(8), 30–37. https://doi.org/10.1109/MC.2009.263

Krasnoshchok, O., & Lamo, Y. (2014). Extended content-boosted matrix factorization algorithm for recommender systems. In *Procedia Computer Science* (Vol. 35, pp. 417–426). https://doi.org/10.1016/j.procs.2014.08.122

Lerche, L., & Jannach, D. (2014). Using Graded Implicit Feedback for Bayesian Personalized Ranking. In *Proceedings of the 8th ACM Conference on Recommender Systems* (pp. 353–356). New York, NY, USA: ACM.

https://doi.org/10.1145/2645710.2645759

Lin, C. J., Kuo, T. T., & Lin, S. De. (2014). A content-based matrix factorization model for recipe recommendation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 8444 LNAI, pp. 560–571). https://doi.org/10.1007/978-3-319-06605-9_46

Lops, P., de Gemmis, M., & Semeraro, G. (2011). Content-based Recommender Systems: State of the Art and Trends. In F. Ricci, L. Rokach, B. Shapira, & P. B. Kantor (Eds.), *Recommender Systems Handbook* (pp. 73–105). Springer US. https://doi.org/10.1007/978-0-387-85820-3_3

Nguyen, J., & Zhu, M. (2013). Content-boosted matrix factorization techniques for recommender systems. *Statistical Analysis and Data Mining*, *6*(4), 286–301. https://doi.org/10.1002/sam.11184

Peska, L. (2017). Linking Content Information with Bayesian Personalized Ranking via Multiple Content Alignments. In *Proceedings of the 28th ACM Conference on Hypertext and Social Media - HT '17* (pp. 175–183). https://doi.org/10.1145/3078714.3078732

Peska, L., & Vojtas, P. (2014). *Recommending for disloyal customers with low consumption rate*. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 8327 LNCS). https://doi.org/10.1007/978-3-319-04298-5_40

Peska, L., & Vojtas, P. (2017). Using Implicit Preference Relations to Improve Recommender Systems. *Journal on Data Semantics*, *6*(1). https://doi.org/10.1007/s13740-016-0061-8

Pilászy, I., & Tikk, D. (2009). Recommending New Movies: Even a Few Ratings Are

More Valuable Than Metadata. In *Proceedings of the Third ACM Conference on Recommender Systems* (pp. 93–100). New York, NY, USA: ACM. https://doi.org/10.1145/1639714.1639731

Qian, N. (1999). On the momentum term in gradient descent learning algorithms. *Neural Networks*. https://doi.org/10.1016/S0893-6080(98)00116-6

Rendle, S., Freudenthaler, C., Gantner, Z., & Schmidt-Thieme, L. (2009). BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence* (pp. 452–461). Arlington, Virginia, United States: AUAI Press.

Ruder, S. (2017). An overview of gradient descent optimization algorithms. *ArXiv:1609.04747*. Retrieved from https://arxiv.org/abs/1609.04747v2

Salakhutdinov, R., & Mnih, A. (2007). Probabilistic Matrix Factorization. *Proc. Advances in Neural Information Processing Systems 20 (NIPS 07)*, 1257–1264. https://doi.org/10.1145/1390156.1390267

Skopal, T., Peška, L., Kovalcík, G., Grošup, T., & Lokoc, J. (2017). Product exploration based on latent visual attributes. In *International Conference on Information and Knowledge Management, Proceedings* (Vol. Part F1318). https://doi.org/10.1145/3132847.3133175

Weimer, M., Karatzoglou, A., & Smola, A. (2008). Improving maximum margin matrix factorization. In *Machine Learning* (Vol. 72, pp. 263–276). https://doi.org/10.1007/s10994-008-5073-7

Yi, X., Hong, L., Zhong, E., Liu, N. N., & Rajan, S. (2014). Beyond Clicks: Dwell Time for Personalization. In *Proceedings of the 8th ACM Conference on Recommender Systems* (pp. 113–120). New York, NY, USA: ACM. https://doi.org/10.1145/2645710.2645724

Zeiler, M. D. (2012). ADADELTA: An Adaptive Learning Rate Method. *ArXiv:1212.5701*. Retrieved from http://arxiv.org/abs/1212.5701

Zhen, Y., Li, W. J., & Yeung, D. Y. (2009). TagiCoFi: Tag Informed Collaborative Filtering. *Proceedings of the Conference on Recommender Systems (RecSys)*, 69–76. https://doi.org/10.1145/1639714.1639727

Zheng, X., Ding, H., Mamitsuka, H., & Zhu, S. (2013). Collaborative matrix factorization with multiple similarities for predicting drug-target interactions. *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '13*, 1025–1033. https://doi.org/10.1145/2487575.2487670