A Latent-Feature Plackett-Luce Model for Dyad Ranking Completion

Dirk Schäfer

dirk.schaefer@jivas.de

Abstract. Dyad ranking is a specific type of preference learning problem, namely the problem of learning a model that is capable of ranking a set of feature vector pairs, called dyads. In this paper a situation is considered where feature vectors consists of identifiers only. A new method based on learning latent-features is introduced for this purpose. The method is evaluated on synthetic data and is applied on the problem of ranking from implicit feedback.

Keywords: Preference Learning, Dyad Ranking, Plackett-Luce Model, Matrix Factorization, Implicit Feedback

1 Introduction

Preference learning is an emerging subfield of machine learning, which deals with the induction of preference models from observed or revealed preference information [3]. Such models are typically used for prediction purposes, for example to predict context-dependent preferences of individuals on various choice alternatives. Depending on the representation of preferences, individuals, alternatives, and contexts, a large variety of preference models are conceivable, and many such models have already been studied in the literature.

A specific problem within the realm of preference learning is the problem of *label ranking*, which consists of learning a model that maps instances to rankings (total orders) over a finite set of predefined alternatives [11]. An instance, which defines the context of the preference relation, is typically characterized in terms of a set of attributes or features; for example, an instance could be a person described by properties such as sex, age, income, etc. As opposed to this, the alternatives to be ranked, e.g., the political parties of a country, are only identified by their name (label), while not being characterized in terms of any properties or features.

In [9], *dyad ranking* is introduced as a practically motivated generalization of the label ranking problem. In dyad ranking, not only the instances but also the alternatives are represented in terms of attributes—a dyad is a pair consisting of an instance and an alternative. Moreover, for learning in the setting of dyad ranking, an extension of the Plackett-Luce model, a statistical model for rank data, was proposed. The approach was based on modeling utility scores of dyads via a bilinear product of feature vectors of instance and alternative respectively.

In: M. Leyer (Ed.): Proceedings of the LWDA 2017 Workshops: KDML, FGWM, IR, and FGDB.
Rostock, Germany, 11.-13. September 2017, published at http://ceur-ws.org

In this paper a problem is addressed which is in a sense diametrical to the above mentioned settings because instances and alternatives are *not* described by any attributes. The reasons of this circumstance could be that observable attributes do not relate well with the preference information or the attributes contain many missing values or they do not exist at all. Data preprocessing and feature engineering may fail in these cases. Often however, instances and alternatives are identified by a unique ID, especially when they are organized in a relational database. Otherwise training examples from a finite set can always be enumerated and a unique ID can be assigned. For this scenario a new variation of the bilinear Plackett-Luce model is introduced which is called *Latent-Feature Plackett-Luce* model (LFPL). In contrast to the original formulation of the bilinear Plackett-Luce model it can deal with rankings over dyads which are specified by identifiers only.

The rest of the paper is organized as follows. First a formal description of the dyad ranking completion problem is given in Section 2. The LFPL method to solve this problem based on latent-features is described in Section 3. The relation between dyad ranking completion and the application of learning to rank on implicit feedback data is described in Section 4. An overview of related methods is provided in Section 5. Experimental results are presented in Section 6, prior to concluding in Section 7.

2 Problem Setting

2.1 Dyad Ranking

The problem of a dyad ranking is to learn a model that accepts as input any set of (new) dyads and produces as output a ranking of them. A dyad is a pair of feature vectors $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{Z} = \mathbb{X} \times \mathbb{Y}$, where the feature vectors are from two (but not necessarily different) domains $\mathbb{X}$ and $\mathbb{Y}$.

Figure 1 shows the basic learning workflow: a learning algorithm creates a model from a finite set of training examples. The trained model can afterwards be used for producing rankings on (new) dyads. The training set $\mathcal{D} = \{\rho_n\}_{n=1}^{N}$ consists of a set examples ρ_n $(1 \leq n \leq N)$, where each example is a dyad ranking

$$\rho_n : \boldsymbol{z}^{(1)} \succ \boldsymbol{z}^{(2)} \succ \ldots \succ \boldsymbol{z}^{(M_n)}, \;\; M_n \geq 2, \tag{1}$$

of length M_n, where M_n can vary from example to example.

2.2 Dyad Ranking Completion

The domain from which the training dyads and their members respectively stem from is denoted by $\mathcal{R} \subset \mathbb{X}$ and $\mathcal{C} \subset \mathbb{Y}$ (indicating (r)ows and (c)columns). Depending on the occurrence of dyad members within the training set one can characterize the dyads which need to be ranked. Table 1 provides an overview of the types of new dyads which can be encountered at the prediction phase. Dyad ranking completion deals with dyads of type 1 which are dyads whose members

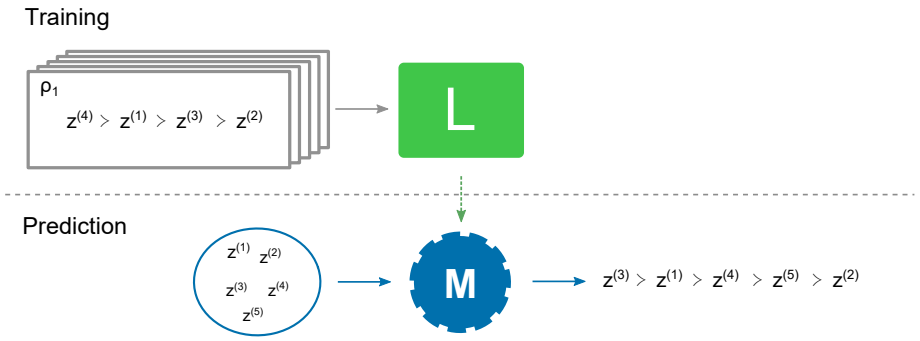

Fig. 1. Generic supervised learning task in dyad ranking.

have been encountered at the training phase individually but not yet jointly together. They define the gaps in the $\mathcal{R} \times \mathcal{C}$ matrix and filling these by ranking all of them or subsets of them characterizes the task.

Table 1. Characterization of dyads depending on whether their members have been encountered in the training phase individually but not conjointly.

Dyad Type	Member 1 Encountered?	Member 2 Encountered?	Prediction Domain
1	yes	yes	$\mathcal{R} \times \mathcal{C}$
2	yes	no	$\mathcal{R} \times \mathbb{Y}$
3	no	yes	$\mathbb{X} \times \mathcal{C}$
4	no	no	$(\mathbb{X} \setminus \mathcal{R}) \times (\mathbb{Y} \setminus \mathcal{C})$

Ranking with Identifiers Dyads whose members are not described by attributes can be utilized either by using existing database IDs or by assigning natural numbers to them. When these dyads are encountered at the training phase then any new dyads in the prediction phase must necessarily be of type 1.

3 A Plackett-Luce Model with Latent Features

3.1 Plackett-Luce Model

The Plackett-Luce (PL) model is a parameterized probability distribution on the set of all rankings over a set of alternatives $y_1, \ldots, y_K$. It is specified by a parameter vector $\boldsymbol{v} = (v_1, v_2, \ldots, v_K) \in \mathbb{R}_+^K$, in which v_i accounts for the (latent)

utility or "skill" of the option y_i. The probability assigned by the PL model to a ranking π is given by

$$\mathbf{P}(\pi \,|\, \boldsymbol{v}) \;=\; \prod_{i=1}^{K} \frac{v_{\pi(i)}}{v_{\pi(i)} + v_{\pi(i+1)} + \ldots + v_{\pi(K)}} = \prod_{i=1}^{K-1} \frac{v_{\pi(i)}}{\sum_{j=i}^{K} v_{\pi(j)}} \;\; . \tag{2}$$

Obviously, the Plackett-Luce model is only determined up to a positive multiplicative constant, i.e., $\mathbf{P}(\pi \,|\, \boldsymbol{v}) \equiv \mathbf{P}(\pi \,|\, s \cdot \boldsymbol{v})$ for all $s > 0$.

As an appealing property of the PL model is that its marginal probabilities (probabilities of rankings on subsets of alternatives) are easy to compute and can be expressed in closed form. More specifically, the marginal of a PL model on $M < K$ alternatives $y_{i(1)}, \ldots, y_{i(M)}$ is again a PL model with parameters $v_{i(1)}, \ldots, v_{i(M)}$.

Bilinear Plackett-Luce Model The bilinear model (BilinPL) [9] is based on a functional formulation of the skill parameters as follows,

$$v(\boldsymbol{z}) = v(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(\boldsymbol{x}^\top \mathbf{W} \boldsymbol{y}\right) \tag{3}$$

that is obtained by defining Φ as the Kronecker (tensor) product:

$$\Phi(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x} \otimes \boldsymbol{y} = \left(x_1 \cdot y_1, x_1 \cdot y_2, \ldots, x_r \cdot y_c\right) = \mathbf{vec}\left(\boldsymbol{x}\boldsymbol{y}^\top\right) \;\; , \tag{4}$$

which is a vector of length $p = r \cdot c$ consisting of all pairwise products of the components of $\boldsymbol{x}$ and $\boldsymbol{y}$, also known as cross-products. Thus, the inner product $\langle \boldsymbol{w}, \Phi(\boldsymbol{x}, \boldsymbol{y}) \rangle$ can be rewritten as a bilinear form $\boldsymbol{x}^\top \mathbf{W} \boldsymbol{y}$ with an $r \times c$ matrix $\mathbf{W} = (w_{i,j})$; the entry $w_{i,j}$ can be considered as the weight of the interaction term $x_i y_j$.

3.2 Latent-Feature based PL Model

To circumvent the limitation of the Bilinear PL model on the dependence of explicit features a novel extension of the Plackett-Luce model is proposed. It is based on latent-features and is called LFPL. For this model the PL log-skill parameter for a dyad $\boldsymbol{z} = (i, j)$ is defined as $\log v(\boldsymbol{z}) = \boldsymbol{U}_i \boldsymbol{V}_j^\top$.

The following notation is used. The indices i and j refers to the entities associated with the first and the second dyad members, whereas $\boldsymbol{U}$ and $\boldsymbol{V}$ are matrices with $\boldsymbol{U} \in \mathbb{R}^{|\mathcal{R}| \times K}$ and $\boldsymbol{V} \in \mathbb{R}^{|\mathcal{C}| \times K}$, where K denotes the dimensionality of latent feature vectors. $|\mathcal{R}|$ and $|\mathcal{C}|$ denote the number of entities in $\mathcal{R}$ and $\mathcal{C}$ respectively. Both, $\mathcal{R}$ and $\mathcal{C}$ refer to special feature spaces which are one dimensional and are made up of natural numbers. The notation $\boldsymbol{U}_k$ refers to the $k-th$ row vector of the matrix $\boldsymbol{U}$, whereas $\boldsymbol{U}_{k,i}$ refers to the $i-th$ latent factor of the $k-th$ vector.

The conditional probability of observing a dyad ranking with the parameters $\boldsymbol{\theta} = \{\boldsymbol{U}, \boldsymbol{V}\}$ can then be stated as:

$$\mathbf{P}(\pi_n \,|\, \varrho_n, \boldsymbol{\theta}) = \prod_{k=1}^{M_n - 1} \frac{\exp\left[\boldsymbol{U}_{z_n(k,1)} \boldsymbol{V}_{z_n(k,2)}^\top\right]}{\sum_{l=k}^{M_n} \exp\left[\boldsymbol{U}_{z_n(l,1)} \boldsymbol{V}_{z_n(l,2)}^\top\right]} \;\; . \tag{5}$$

The function $z_n(k,i)$ in the likelihood (5) refers to the $i-th$ member of a dyad $\boldsymbol{z}$ observed in sample ϱ_n, where $i \in \{1,2\}$. The index k refers to the dyad $\boldsymbol{z}_n^{(j)}$ which is put at the $k-th$ rank of π_n, s.t. $\pi_n(k) = j$.

The model parameters $\boldsymbol{\theta} = \{\boldsymbol{U}, \boldsymbol{V}\}$ can be found by minimizing the negative log-likelihood (NLL) of the data,

$$\ell = \sum_{n=1}^{N} \ell_n \ . \tag{6}$$

In (6) the NLL of a single example (dyad ranking) is given by

$$\ell_n = -\log P(\pi_n \mid \varrho_n, \theta) \tag{7}$$

$$= \sum_{k=1}^{M_n-1} \log \left(\sum_{l=k}^{M_n} \exp \left[\boldsymbol{U}_{z_n(l,1)} \boldsymbol{V}_{z_n(l,2)}^{\top} \right] \right) - \sum_{k=1}^{M_n-1} \boldsymbol{U}_{z_n(k,1)} \boldsymbol{V}_{z_n(k,2)}^{\top} \ . \tag{8}$$

The overall objective which is to be optimized is hence of the form:

$$\ell(\boldsymbol{\theta}, \mathcal{D}) + \Omega(\boldsymbol{\theta}) \ , \tag{9}$$

where $\Omega(\boldsymbol{\theta})$ is a regularization term for controlling the model capacity or likewise a countermeasure for preventing over-fitting. Note that the objective (9) is not convex and its minimization requires special considerations.

Learning Algorithm As learning algorithm the first order optimization technique *alternating gradient descent* is used. With this approach the model weights are updated by keeping one matrix $\boldsymbol{U}$ or $\boldsymbol{V}$ fixed respectively. The first derivatives are given by

$$\frac{\partial \ell_n}{\partial \boldsymbol{U}_{s,i}} = \sum_{k=1}^{M_n-1} \frac{\sum_{l=k}^{M_n} \mathbf{1}_{\{s=z_n(l,1)\}} \boldsymbol{V}_{z_n(l,2),i} \cdot \exp \left[\boldsymbol{U}_{z_n(l,1)} \boldsymbol{V}_{z_n(l,2)}^{\top} \right]}{\sum_{l=k}^{M_n} \exp \left[\boldsymbol{U}_{z_n(l,1)} \boldsymbol{V}_{z_n(l,2)}^{\top} \right]} - \sum_{k=1}^{M_n-1} \mathbf{1}_{\{s=z_n(k,1)\}} \boldsymbol{V}_{z_n(k,2),i} \ , \tag{10}$$

with $s \in \mathcal{R}_n$ and $1 \leq i \leq K$. Here $\mathcal{R}_n$ refers to the set of entities represented by numerical IDs that occur as *first* dyad members in the sample n. For example, for a dyad ranking $\rho_n : (x_3, y_2) \succ (x_3, y_{144}) \succ (x_5, y_9)$, $\mathcal{R}_n$ would be $\{3,5\}$ and $\mathcal{C}_n = \{2,9,144\}$. The derivatives for $\boldsymbol{V}$ can be stated similarly by treating the components of $\boldsymbol{U}$ as constants:

$$\frac{\partial \ell_n}{\partial \boldsymbol{V}_{t,i}} = \sum_{k=1}^{M_n-1} \frac{\sum_{l=k}^{M_n} \mathbf{1}_{\{t=z_n(l,2)\}} \boldsymbol{U}_{z_n(l,1),i} \cdot \exp \left[\boldsymbol{U}_{z_n(l,1)} \boldsymbol{V}_{z_n(l,2)}^{\top} \right]}{\sum_{l=k}^{M_n} \exp \left[\boldsymbol{U}_{z_n(l,1)} \boldsymbol{V}_{z_n(l,2)}^{\top} \right]} - \sum_{k=1}^{M_n-1} \mathbf{1}_{\{t=z_n(k,2)\}} \boldsymbol{U}_{z_n(k,1),i} \ , \tag{11}$$

with $t \in \mathcal{C}_n$ and $1 \leq i \leq K$.

Algorithm 1 uses these expressions together with the AdaGrad strategy for adaptive learning rate updates[1]. AdaGrad provides the advantage of freeing the user from determining the initial learning rate values and update schedules while being simple to implement [2]. The actual implementation utilizes a quadratic regularization term

$$\Omega(\boldsymbol{\theta}) = \frac{1}{2}\lambda(\|\boldsymbol{U}\|_F^2 + \|\boldsymbol{V}\|_F^2) \ , \tag{12}$$

where $\|\cdot\|_F$ refers to the Frobenius matrix norm. This approach is theoretically justified by considering it as a problem of learning low rank matrices with a convex relaxation; a relaxation that is based on the nuclear (trace) norm regularization [1].

Algorithm 1 LFPL Alternating Gradient Descent with AdaGrad

Require: Initial matrices $\boldsymbol{U} \in \mathbb{R}^{|\mathcal{R}|\times K}$, $\boldsymbol{V} \in \mathbb{R}^{|\mathcal{C}|\times K}$, learning rate γ, number of iterations T, dyad rankings $\mathcal{D}_{Tr} = \{(\varrho, \pi)_n\}$.

function TRAIN LFPL ($\mathcal{D}_{Tr}$, $\boldsymbol{U}$, $\boldsymbol{V}$, T)
  for $i \leftarrow 1$ to T **do**

    $\nabla_U \leftarrow$ **Formula (10)** on all samples $(\varrho, \pi)_n$ $1 \leq n \leq N$
    $\boldsymbol{H}_U = \boldsymbol{H}_U + (\nabla_U)^2$
    $\boldsymbol{U} \leftarrow \boldsymbol{U} - \gamma(\boldsymbol{H}_U)^{-1/2} \odot \nabla_U$

    $\nabla_V \leftarrow$ **Formula (11)** on all samples $(\varrho, \pi)_n$ $1 \leq n \leq N$
    $\boldsymbol{H}_V = \boldsymbol{H}_V + (\nabla_V)^2$
    $\boldsymbol{V} \leftarrow \boldsymbol{V} - \gamma(\boldsymbol{H}_V)^{-1/2} \odot \nabla_V$
  end for
  return $(\boldsymbol{U}, \boldsymbol{V})$
end function

3.3 Predictions with LFPL

With LFPL it is possible to solve the dyad ranking completion problem from Section 2.2 by making prediction involving dyads of type 1 (c.f. Table 1). They are all of the form $z^{(k)} = (i_k, j_k)$, where $i_k, j_k \in \mathbb{N}$ and both indices i_k and j_k appeared independently at other dyads during training but not jointly together yet. The most probable ranking can be predicted by calculating $v^{(k)} = \exp(\boldsymbol{U}_{i_k}\boldsymbol{V}_{j_k}^\top)$ and then arranging the dyads in descending order to the scores.

[1] Note, that any other learning rate update strategy for gradient descent could be used too.

3.4 Connection with the Bilinear PL Model

There are two ways to relate the LFPL model to the Bilinear PL model. The first relation can be seen by setting $\boldsymbol{W} = \boldsymbol{U}\boldsymbol{V}^\top$. By describing each entity i by means of a label embedding $\boldsymbol{e}_i$ (e.g., via 1-of-k encoding), it is possible to cast (5) to the Bilinear PL model formulation (3). The second relation can be seen by treating the latent feature vectors in the rows of $\boldsymbol{U}$ and $\boldsymbol{V}$ as object features and the bilinear weight matrix $\boldsymbol{W}$ as identity matrix $\boldsymbol{I}_K$. Both relations justify to classify the LFPL to be a bilinear model, too. But in contrast to BilinPL (Section 3.1) the LFPL model does not require the engineering of explicit features but learns latent-feature vectors under the same bi-linearity assumption instead. Without considering $\boldsymbol{W}$ to be of lower rank it is not possible for the BilinPL approach to generalize to unseen dyad rankings of type 1 where dyad members are described by identifiers only.

4 Learning from Implicit Feedback

Implicit feedback is probably the most basic kind of preference statement. It refers to the selection of a single item from a set of choice alternatives at a time. A reasonable interpretation for that action is that the selected item is probably preferred over other items present at the time of selection. This kind of data can be represented in a rectangular schema of rows and columns, e.g., where they represent users and items. Cells in that schema indicate if an interaction between a particular row and a column happened. In what follows we denote row indices as $i_k \in \mathcal{R}$ and column indices as $j_l \in \mathcal{C}$. Formally, implicit feedback data is defined as the subset of indices $\mathcal{F} \subset \mathcal{R} \times \mathcal{C}$. An index pair (i, j) in $\mathcal{F}$ can be interpreted as an action that happened between entities i and j. The goal is to provide a ranking of dyads (i, j) for which no interactions have been observed so far, i.e., $(i, j) \notin \mathcal{F}$.

Without the loss of generality we turn now to the special case of the LFPL for *contextual* implicit feedback. Let the set of items on which interactions have been recorded for a context i be denoted as $C_i^+ := \{j \in \mathcal{C} \,|\, (i, j) \in \mathcal{F}\}$. We treat this scenario as a completion problem which we want to solve with dyad ranking. Figure 2 illustrates the conversion of original implicit feedback data to contextual dyad rankings. For each row i_k (or user) a separate dyadic preference graph is generated. All graphs combined form a training data set $D_\mathcal{F}$, which is

$$D_\mathcal{F} = \{(i, j_a) \succ (i, j_b) \,|\, j_a \in C_i^+ \ \wedge \ j_b \in \mathcal{F} \setminus \mathcal{C}_i^+\} \ .$$

To overcome the problem of over-fitting on high frequent items, the learning procedure needs a slight modification. The new procedure uses sampling from $D_\mathcal{F}$ similar to BPRMF [7] but reuses the existing Algorithm 1 for this purpose. Instead of acting on single training examples it generates subsets of examples (batches) of size L. This new approach is called LFPL/IF (for implicit feedback) and is stated as Algorithm 2. As stopping criterion either the convergence of the performance on a hold-out set or a predefined maximal number of iterations can be used.

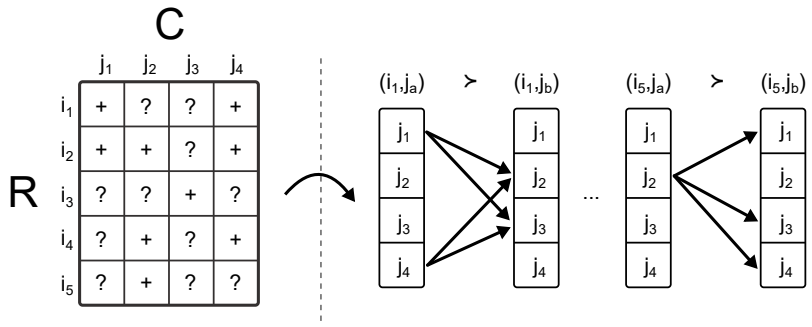

Fig. 2. Conversion of implicit feedback data (left side) to dyad rankings (right side).

Algorithm 2 LFPL/IF Learning Algorithm

Require: Implicit feedback data $D_{\mathcal{F}}$, number of latent factors K, batch-size L
1: Initialize $\boldsymbol{U} \in \mathbb{R}^{|\mathcal{R}| \times K}$, $\boldsymbol{V} \in \mathbb{R}^{|\mathcal{C}| \times K}$
2: **repeat**
3: $\quad \mathcal{S} \leftarrow$ draw L many dyadic preferences from $D_{\mathcal{F}}$
4: $\quad$ **Update LFPL** $(\mathcal{S}, \boldsymbol{U}, \boldsymbol{V}, 1)$ ▷ Algorithm 1
5: **until Stopping Criterion**
6: **return** $\boldsymbol{U}$, $\boldsymbol{V}$

Gradients of LFPL/IF The calculations of the gradient matrices ∇_U in Eq. (10) and ∇_V in Eq. (11) of Algorithm 1 simplify drastically in the case of implicit feedback data. Let j_a, j_b denote the indices of items, where the corresponding item j_a is preferred over j_b. Let furthermore i be the index of a user and $1 \leq k \leq K$ be a latent feature dimension. The NLL of a training example is then given by

$$\ell_n = \log(\exp(U_i V_{j_a}) + \exp(U_i V_{j_b})) - U_s V_{j_a} \ .$$

And the derivatives can be calculated using following expressions:

$$\frac{\partial \ell_n}{\partial U_{i,k}} = \frac{V_{j_a,k} \exp(U_i V_{j_a}) + V_{j_b,k} \exp(U_i V_{j_b})}{\exp(U_i V_{j_a}) + \exp(U_i V_{j_b})} - V_{j_a,k} \ , \tag{13}$$

$$\frac{\partial \ell_n}{\partial V_{j_a,k}} = \frac{U_{i,k} \exp(U_i V_{j_a})}{\exp(U_i V_{j_a}) + \exp(U_i V_{j_b})} - U_{s,k} \ , \tag{14}$$

$$\frac{\partial \ell_n}{\partial V_{j_b,k}} = \frac{U_{i,k} \exp(U_i V_{j_b})}{\exp(U_i V_{j_a}) + \exp(U_i V_{j_b})} \ . \tag{15}$$

5 Related Work

Plackett-Luce Networks (PLNet) is a method for dyad ranking that is able to learn joint-feature representations [10]. Being based on a neural network it is

possible to express non-linear relationships between dyads and rankings. It is applicable of solving the dyad ranking completion problem too but it is not designed for the special case where dyad inputs are expressed by identifiers only.

Probabilistic Matrix Factorization (PMF) [8] is a classical preference completion method for preferences in form of *ratings*. A substantial property of PMF is the learning and the prediction of *quantitative* instead of *qualitative* preferences by filling gaps of an incomplete rating matrix. Although LFPL and PMF differ fundamentally in this point, they also share some aspects. Firstly, both approaches are based on the same model class, i.e., matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ are multiplied together to give a real valued matrix of dimensionality $N \times M$. In both cases, higher matrix values indicate stronger preferences. Secondly, both approaches put Gaussian priors over $\boldsymbol{U}$ and $\boldsymbol{V}$ to control model capacity.

The methods BPRMF and WRMF are standard methods for learning on implicit feedback data. The LFPL method is closely related to BPRMF (the matrix factorization variant of Bayesian Personalized Ranking) although the first acts on dyad rankings of the form $(i, j_a) \succ (i, j_b)$ and the latter on triplets $j_a \succ_i j_b$ [7]. Both approaches share the same model class and also the assumption of Gaussian priors. LFPL generalizes BPRMF by supporting rankings of arbitrary lengths instead of pairwise preferences, i.e., rankings of length 2, and by providing contextual and non-contextual variants. WRMF is an approach that is also based on matrix factorization [6]. It shares with LFPF and BPRMF the same model class but in contrast to them it is a pointwise approach (one item) instead of a pairwise approach (two items at a time).

6 Experiments

Ranking Completion (Synthetic Data) The advantage of LFPL over BilinPL and PLNet is shown for the problem of ranking completion with identifiers. Given a set of row entities $\mathcal{R}$ and a set of column entities $\mathcal{C}$, which are specified by their IDs only. Furthermore, provided with a training set of incomplete contextualized rankings over $\mathcal{C}$, where contexts are provided by entities from $\mathcal{R}$, the task is to create a model which can be used to predict arbitrary rankings for a set of new dyads in $\mathcal{R} \times \mathcal{C}$.

One thousand rankings of lengths 10 were sampled from the LFPL model distribution. In a 10-fold CV experiment where 900 of them were used for training and 100 for testing for each run. The LFPL model used for the generation of rankings was determined by the matrices $\boldsymbol{U} \in \mathbb{R}^{|\mathcal{R}| \times K}$ and $\boldsymbol{V} \in \mathbb{R}^{|\mathcal{C}| \times K}$ which were generated by sampling from the normal distribution, with $K = 5$, the number of latent factors. To apply the dyads with identifiers on BilinPL and PLNet the natural numbers used as identifiers were converted into vector format using 1-of-k encoding. PLNet is configured to contain one hidden layer with 10 neurons. The results given in Table 2 confirmed the hypotheses that BilinPL and PLNet do not generalize well beyond the observed preferences. Especially PLNet is prone to overfitting. LFPL in contrast to BilinPL is able to *share* model

parameters under the assumption that preferences are representable with a lower rank matrix W.

Table 2. Results of the synthetic preference completion problem. Performance values in terms of Kendall's τ are obtained via 10-fold CV on 1000 rankings, $|X_i| = 10$.

$\lvert\mathcal{R}\rvert \times \lvert\mathcal{C}\rvert$	40×50	60×100	80×150
Completeness (%)	2.026	0.225	0.056
BilinPL	.844 ± .025	.724 ± .018	.770 ± .023
LFPL	.973 ± .006	.950 ± .014	.923 ± .009
PLNet	.721 ± .023	.611 ± .031	.493 ± .032

Implicit Feedback (Real Data) In this experiment the capabilities of LFPL/IF on implicit feedback data was tested. The data set Movielense (1M) was utilized due to its availability and high degree of popularity [5]. It is about 6040 users which provided a total amount of 1 million explicit ratings on 3900 movies. Similar to the experimental setup of [7] the rating data set was converted into implicit feedback data. Each rating $1 \leq r \leq 5$ is interpreted as a feedback and the task corresponds to estimating how likely a user would rate the residual entries.

For the evaluation we repeated the following leave-one-out procedure: One random rating entry per user is removed from the data. These entries are not considered for training but for benchmarking the trained model afterwards. Training and test sets are thus disjunct subsets of user associated item indices, i.e., $C_i^{Tr}, C_i^{Te} \subset C_i^+$, $1 \leq i \leq |U|$ and $C_i^{Tr} \cap C_i^{Te} = \emptyset$. For evaluation of the models the average AUC is used:

$$AUC = \frac{1}{|U|} \sum_u \frac{1}{|\mathcal{E}(u)|} \sum_{(j_a, j_b) \in \mathcal{E}(u)} I[\hat{v}(u, j_a) > \hat{v}(u, j_b)] \ ,$$

in which the evaluation pairs $\mathcal{E}(u)$ are given by

$$\mathcal{E}(u) = \left\{ (j_a, j_b) \,|\, j_a \in C_u^{Tr} \wedge j_b \notin (C_u^{Tr} \cup C_u^{Te}) \right\} \ .$$

As baseline methods random guessing and the user-independent most popular approach are used. Most popular determines $\hat{v}(u, i)$ as the number of users that interacted with item i. The implementations of the BPRMF and WRMF methods were taken from the software package MyMediaLite 3.11 [4] and their default parameters were adopted. For LFPL the parameters were: dimension of latent feature space set to 10, regularization values were fixed to 0.01, number of iterations was 5000 and the batch size was set to 2000.

The results provided in Figure 3 reflect my expectation that LFPL/IF performs similarly as BPRMF and WRMF. All three methods outperform the base-

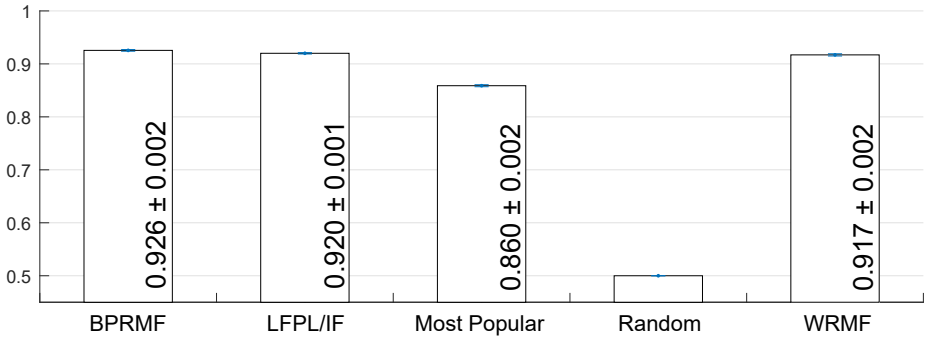

Fig. 3. Results of the Movielense 1M Implicit Feedback experiment.

lines methods to a large extent. In direct comparison LFPL/IF is slightly superior to WRMF and inferior to BPRMF in terms of AUC performance.

7 Conclusion

We presented a new dyad ranking approach based on the Plackett-Luce model with latent-feature vectors applicable for the dyad ranking completion problem. The advantage of LFPL is its applicability in scenarios where no information on dyad members are available but their unique identifiers. The problem of learning and prediction given implicit feedback data could be addressed as an application of dyad ranking completion. The learning algorithm is based on alternating gradient descent to deal with the non-convexity of the objective function. An obvious disadvantage is that predictions with latent features can be carried out in a straightforward way only on dyads within $\mathcal{R} \times \mathcal{C}$. LFPL in its current form is also a bilinear model and may be limited to model more complicated non-linear relationships between dyads and preferences.

References

1. Carlo Ciliberto, Dimitris Stamos, and Massimiliano Pontil. Reexamining low rank matrix factorization for trace norm regularization. *CoRR*, abs/1706.08934, 2017.
2. John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.
3. Johannes Fürnkranz and Eyke Hüllermeier. Preference learning: An introduction. In Johannes Fürnkranz and Eyke Hüllermeier, editors, *Preference Learning*, pages 1–17. Springer-Verlag, 2010.
4. Zeno Gantner, Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. MyMediaLite: A free recommender system library. In *Proceedings of the 5th ACM Conference on Recommender Systems (RecSys 2011)*, 2011.
5. F. Maxwell Harper and Joseph A. Konstan. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4):19:1–19:19, 2015.

6. Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, pages 263–272. IEEE Computer Society, 2008.
7. Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th conference on uncertainty in artificial intelligence*, pages 452–461. AUAI Press, 2009.
8. Ruslan Salakhutdinov and Andriy Mnih. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems*, volume 20, 2008.
9. Dirk Schäfer and Eyke Hüllermeier. Dyad ranking using a bilinear Plackett-Luce model. In *Proceedings ECML/PKDD–2015, European Conference on Machine Learning and Knowledge Discovery in Databases*, Porto, Portugal, 2015. Springer.
10. Dirk Schäfer and Eyke Hüllermeier. Plackett-Luce networks for dyad ranking. In *Workshop LWDA, "Lernen, Wissen, Daten, Analysen"*, Potsdam, Germany, 2016.
11. S. Vembu and T. Gärtner. Label ranking algorithms: A survey. In J. Fürnkranz and E. Hüllermeier, editors, *Preference Learning*. Springer-Verlag, 2011.