First-order Probabilistic Model for Hybrid Recommendations

Julia Hoxha
Karlsruhe Institute of Technology
Karlsruhe, Germany
Email: julia.hoxha@kit.edu

Achim Rettinger
Karlsruhe Institute of Technology
Karlsruhe, Germany
Email: achim.rettinger@kit.edu

***Abstract*—In this paper, we address the task of inferring user preference relationships about various objects in order to generate relevant recommendations. The majority of the traditional approaches to the problem assume a flat representation of the data, and focus on a single dyadic relationship between the objects.**

We present a richer theoretical model for making recommendations that allows us to reason about many different relations at the same time. The model is based on Markov logic, which is a simple and powerful language that combines first-order logic and probabilistic graphical models. We apply a hybrid, content-collaborative merging scheme through feature combination. We experimentally verify the efficacy of our theoretical model, and show that our method outperforms state-of-the-art recommendation approaches.

***Keywords*—hybrid recommender; markov logic; rating prediction; first-order probabilistic logic; markov logic networks; recommendation system;**

I. Introduction

Extensive work has been done in the field of recommender systems to make use of the enormous online information of user activities for inferring user preference relationships about various products, books, web pages, or other information, which we generically refer to as *objects*. In the recommendation task, we are interested in predicting how likely a user is interested in a particular object, given information about this user, the other users' historical behavior, and information about the objects.

Traditional approaches to the problem derive from classical algorithms in statistical pattern recognition and machine learning. The majority of these approaches assume a *flat* data representation for each object, and focus on a single dyadic relationship between the objects. In web usage analysis, for example, the information sources might include user access logs, the relationships between the web pages visited, reviews written by the user, meta-data on the site and additional information about the user. These information can be aggregated in an e-commerce setting, where we include customers buying patterns to make predictions about future purchases. In the context of the World Wide Web, where there is often much more relational information available than a single user-item relationship, we need added modeling power to capture richer relational information.

In this paper, we examine a richer model for making recommendations that allows us to reason about many different relations at the same time. It takes advantage of the recent progress in statistical relational learning (a.k.a. multi-relational data mining), which provides rich representations and efficient inference and learning algorithms for non-i.i.d. data [4]. In particular, we use Markov logic, which combines first-order logic and Markov random fields [11], resulting in the refined probabilistic models of Markov Logic Networks (MLNs), which have emerged as a powerful and popular framework combining logical and probabilistic reasoning.

A key advantage of MLNs is that they allow to express semantically-rich formulae to capture a variety of dependencies between entities in a seamless fashion. MLNs can be used for reasoning about an entity using the entire rich structure of knowledge encoded by the relational representation. The first-order probabilistic model that we propose makes it possible to combine many different objects and relations into a comprehensive solution to the recommendation task. We deploy a *hybrid* approach for generating recommendations, based on a content/collaborative merging scheme through feature combination. The feature-combination hybrid recommender takes into consideration collaborative data, but does not rely on it exclusively. In addition, it considers the information about the inherent similarity of items that are otherwise opaque to a collaborative system.

To the best of our knowledge, this is the first work applying markov logic for the task of recommendation, particularly in a hybrid approach. The proposed theoretical model is generic and allows to model any domain of interest. For clarity, we illustrate in this paper an example of our model in the book-rating domain. We apply efficient methods for inference and parameter learning, as well as highlighting the rich modeling power in addressing real-world recommendation tasks. In addition, we experimentally verify the efficacy of our theoretical model for making recommendations.

II. Related Work

The general recommendation problem is built on the user-item matrix R of U users and I items, where the element r_{ij} is the rating given by user u to item i. In the matrix, a large scale of ratings are missing. Thus the recommendation task is formalized to predict the missing values in the matrix. The techniques are divided into content-based methods [8] and collaborating filtering (CF) methods [7], [16].

There has been a plethora of approaches introduced in the recommender systems field, but the factorization-based

method, as a kind of collaborative filtering methods, has been demonstrated as most successful in performing the recommendation task with large-scale datasets [1], [5]. A competitive representative of one of the state-of-the-art approaches is the probabilistic matrix factorization (PMF) model [12].

There is another line of works based on relational learning to analyze the probabilistic constraints between the attributes of entities and relationships. Xu et al. [15] extend the expressiveness of relational models by introducing for each entity (or object) an infinite dimensional latent variable as part of a Dirichlet process (DP) mixture model. In an earlier work, Getoor et al. [3] present a conceptual model that allows one to reason about many different relations in a domain based on probabilistic relational models (PRMs). Yet, this work remains conceptual in describing how PRMs can be applied to CF, and its efficacy is not experimentally verified.

Our work differentiates from the existing approaches by introducing a rich theoretical model that is able to capture content-based, as well as CF-based aspects in a recommendation domain. We believe this is also the first work to apply first-order logic to formulate such constraints. The approach is also implemented and experimentally evaluated.

III. Background

Markov Networks. A Markov network (also known as Markov random field) is a model for the joint distribution of a set of variables $X = (X_1, X_2, ..., X_n)$. It is composed of an undirected graph G and a set of potential functions ϕ_k. The graph contains a node for each variable, and the model has a potential function for each clique in the graph. The joint distribution is $P(X = x) = \frac{1}{Z}\prod_x \phi_k(x_{\{k\}})$, where $x_{\{k\}}$ is the state of the the variables that appear in the k-th clique, and Z is the partition function $Z = \sum_x \prod_x \phi_k(x_{\{k\}})$.

Provided that $\forall x, P(X = x) > 0$, Markov networks are also conveniently represented as log-linear models $P(X = x) = \frac{1}{Z}\exp(\sum_x w_i f_i(x))$. Each clique potential is replaced by an exponentiated weighted sum of features of the state. There is one feature, which may be any real-valued function of the state, corresponding to each possible state $x_{\{k\}}$ of each clique, with its weight being log. In such probabilistic models, the goal is to find the most likely state of a set of query (unobserved) variables given the state of a set of evidence variables (this is inference), and to compute the conditional probabilities of unobserved variables (marginal inference).

First-Order Logic A *first-order knowledge base* (KB) is a set of sentences or formulae in first-order logic. First Order Logic (FOL) formulae are composed of four types of symbols: constants, variables, functions and predicates. Constants represent objects in a domain of interest (e.g. people: $Sara$, Bob, etc.). Variable symbols range over the objects. Predicate symbols represent relations betwen objects (e.g. `hasRating`) or attributes of objects (`hasAge`). Variables and constants may be typed, in which case variables only range over objects of the given type. A *term* is any expression representing an object in the domain. It can be a constant, a variable, or a function applied to a tuple of terms.

An atomic formula or *atom* is a predicate symbol applied to a list of terms (e.g. `hasRated`$(Anna, book_1, 5)$. A term is ground when it contains no variables, but all its arguments are constants. A ground atom or ground predicate is an atomic formula all of whose arguments are ground terms. Formulas are recursively constructed from atomic formulas using logical connectives and quantifiers. A possible world is an assignment of truth values to all possible ground atoms.

Markov Logic Networks In first-order logic, the formulae are hard constraints on the set of possible worlds: if a world violates even one formula, it has zero probability. In many applications, there is a need to soften these constraints: when a world violates one formula in the KB it is less probable, but not impossible. This is also the setting that Markov Logic Networks provides. Each formula has an associated weight that reflects how strong is this constraint.

A Markov logic network (MLN) [11] L is a set of pairs (F_i, w_i), where F_i is a formula in first-order logic and w_i is a real number. Given a set of constants C, it defines a Markov Network $M_{L,C}$ as follows:
- $M_{L,C}$ contains one binary node for each possible grounding of each predicate appearing in L. The value of the node is 1 if the ground predicate is true, and 0 otherwise.
- $M_{L,C}$ contains one feature for each possible grounding of each formula $F_i \in L$. The value of this feature is 1 if the ground formula is true, and 0 otherwise. The weight of the feature is the w_i associated with F_i in L.

There is an edge between two nodes in $M_{L,C}$, iff the corresponding ground predicates appear together in at least one grounding of one formula in L. An MLN serves as a template for constructing Markov networks. The probability distribution over the possible worlds is: $P(X = x) = \frac{1}{Z}\exp(\sum_{i=1}^{F} w_i n_i(x)) = \frac{1}{Z}\prod_{i=1}^{F} \phi_i(x_{\{i\}})^{n_i(x)}$ where F is the number formulas in the MLN, $n_i(x)$ is the number of true groundings of F_i in x, and $\phi_i(x_{\{i\}}) = e^{w_i}$.

IV. hMLN Probabilistic Model

A. Problem Statement

In our approach, the task of generating recommendations consists in predicting the probability of the existence of a relation r^{ij} between user u_i and object o_j (e.g. $likes(u_1, page_1)$ or $rates(u_1, book_1, 5)$), and then choosing as recommendations the set of objects with the highest probability value.

Let's consider an example of a book rating domain as illustrated in Fig. 1. Each user, with attributes such as address and age, expresses own preferential feedback on objects (in this case books) via ratings. User U_1 has rated $Book_1$ with a score of 5. At the same time, she also assigns a tag with annotation $romantic$ to this book. The book has for author another object, which is from the country US.

Similar relations are also occuring for user U_2, who also rates $Book_1$ with score 5. The task is now to predict which score he would give for $Book_2$, considering the ratings similarities of this user to the other users (i.e. collaborative-filtering features), as well as the attributes of the book and the attributes of the user (i.e. content-based features). We are also interested to consider relations of this user to other objects, e.g. the tags assigned and how they are similar to those of other users.

We present a model, referred here as hMLN, which is able to capture these relationships in a richer way, yet be able to

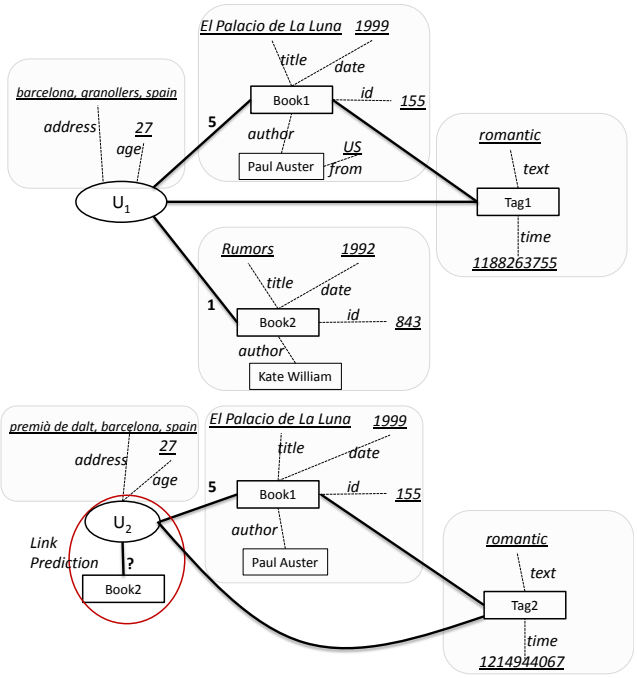

Fig. 1. Example of the prediction task in a relational model

yield correct prediction values of the missing relations (in this case rating of user U_2 for $Book_2$). Based on this task formulation, we present a model for the relationships between user and objects using markov logic. The model is generic and can be applied to any domain. For illustration of the model, we continue to focus on the book-rating domain.

B. Semantics of hMLN

Conceptually, the model consists of three parts: (1) the MLN program that contains the predicates and the first-order logic formuale, (2) the evidence set, and (3) the query set. The evidence set, used for the training, is a list of ground atoms that are deemed to be true unless preceded by "!". The query set is the testing set, which consists of atoms whose arguments are variables. In our case, these define the relations we need to predict.

1) Predicate Schema: In order to model the objects and relations in a domain, we first need to define the predicate schema. The schema consists of a list of predicate declarations. Each predicate declaration specifies a predicate name with a list of argument types. Each type is supported by a set of constants. We distinguish between the following predicates:

- **Object-Declaration Predicate**
  $\texttt{Object}_\texttt{i}(o_i)$
  e.g. $\texttt{User}(person)$, $\texttt{Book}(book)$
  with evidence such as $\texttt{User}(\text{"}Anna\text{"})$, etc.
- **Object-Attributes Predicates**
  $*\texttt{hasAttribute}_\texttt{i}(object_k, object_l)$
  or $*\texttt{hasAttribute}_\texttt{i}(object_k, c)$, where c is a constant.
  For example,
  $*\texttt{hasAge}(person, age)$
  $*\texttt{hasAuthor}(book, author)$
  $*\texttt{hasCountry}(person, country)$
  The attribute can be a literal or another object.
- **User-Object Preference Relations**
  $\texttt{hasRelation}_\texttt{i}(person, object, score)$
  For example: $\texttt{hasRating}(person, book, rating)$
  with an evidence like
  $\texttt{hasRating}(\text{"}Anna\text{"}, \text{"}book_2\text{"}, 2)$.
  If the preference relation has a score other than binary, then a predicate with three arguments is defined. Instead of integer rating, we define levels of preference scores, e.g. using the distribution:
  L1={0-2},L2={3-4},L3={5-6},L4={7-8},L5={9-10}.

  Other examples of user-preference relations are: `tagged`, `visited`, `liked`, `purchased`, etc. A predicate definition preceded by "*" is considered as closed world assumption, i.e. all its ground atoms not listed in the evidence are false.
- **Recommendation Features Predicates**
  To model the dependency between objects we define a predicate, referred to as feature predicate:
  $\texttt{shareFeature}_\texttt{i}(object_k, object_l)$
  These features of dependencies may be qualitative or logical, which define if the relationship exists or not (e.g. $\texttt{sharePublicationDate}(book_k, book_l)$).
  We distinguish between `ObjectFeature` and `UserFeature` features, for example:
  $\texttt{shareAuthor}(\text{"}book_1\text{"}, \text{"}book_2\text{"})$: is a logical object feature, the arguments are instances of objects.
  $\texttt{shareCountry}(\text{"}user_1\text{"}, \text{"}user_2\text{"})$: is a logical user feature, the arguments are instances of users.
  $\texttt{shareAge}(\text{"}user_1\text{"}, \text{"}user_2\text{"})$: a logical user feature.
  Figure I gives a sample of the model with an example from the book rating domain [1].
- **Identity Predicates**
  $\texttt{sameObject}_\texttt{i}(object_k, object_k)$,
  $\texttt{sameBook}(book, book)$
  For exampe, $\texttt{sameUser}(person, person)$
  with an evidence like $\texttt{sameUser}(\text{"}Anna\text{"}, \text{"}Anna\text{"})$.
- **Query Predicates**
  This is the preference relation, whose probability needs to be predicted. The score may be the rating value.
  $\texttt{query\_relation}(person, object, score)$
  For example, $\texttt{rates}(person, book, rating)$
  with an evidence like $\texttt{rates}(\text{"}Bob\text{"}, \text{"}book_1\text{"}, L_5)$.

In order to generate the evidence dataset, we populate the mentioned predicates with instantiations of the objects (i.e. information on users, books, and ratings).

2) Hybrid Recommendation Formulae: A crucial part of MNLs is the set of formulae defined to model the dependencies between objects in the domain of interest. As explained earlier, the formulae (also referred to as *rules*) can be defined as hard or soft, and each has a particular weight.

[1]The goal is to find the highest probable rating ($r \in \{1, ..., 5\}$) that *Bob* gives to $book_1$. We define the schema as a list of predicate declarations. As evidence we are given profile information, as well as known ratings of *Bob* and other users. Any variable not explicitly quantified is universally quantified.

We define the following formulae:

Features Formulae: Modeled as hard rules, these formuale reflect the dependency of objects based on the attributes that they have in common.

Formula F.1

$$\texttt{hasAttribute}_i(o_1, a) \wedge \texttt{hasAttribute}_i(o_2, a) \wedge \texttt{sameObject}_k(a, a) \wedge \neg\texttt{sameObject}_1(o_1, o_2) \Rightarrow \texttt{shareFeature}_i(o_1, o_2)$$

For example, in our running scenario we would have the following formula to express the feature `shareAuthor` between any two objects of type `Book` that have the same `Author` in common:

$$\texttt{hasAuthor}(book_1, auth_1) \wedge \texttt{hasAuthor}(book_2, auth_1) \wedge \texttt{sameAuthor}(auth_1, auth_1) \wedge \neg\texttt{sameBook}(book_1, book_2) \Rightarrow \texttt{shareAuthor}(book_1, book_2)$$

Content-based Dependency Formulae: These are rules that express content-based dependency between the score of the relation that we want to predict and the features of the objects. These are soft rules, whose weight we learn with parameter learning methods (Sec. IV-D).

Formula F.2

$$w_2\ \texttt{hasRelation}_j(u, o_1, r) \wedge \texttt{shareObjectFeature}_i(o_1, o_2) \wedge \texttt{sameFeature}_k(r, r) \Rightarrow \texttt{query\_relation}_j(u, o_2, r)$$

In our example, we would have the following rule:

$$0.8\ \texttt{hasRating}(user, book_1, r) \wedge \texttt{shareAuthor}(book_1, book_2) \wedge \texttt{sameRating}(r, r) \Rightarrow \texttt{rates}(user, book_2, r)$$

Collaborative-filtering Formulae: These rules reflect the similarity of behavior between user features and their rating behavior/preferences.

Formula F.3

$$w_3\ \texttt{hasRelation}_j(u_1, o_1, r) \wedge \texttt{shareUserFeature}_i(u_1, u_2) \wedge \texttt{sameFeature}_k(r, r) \Rightarrow \texttt{query\_relation}_j(u_2, o_1, r)$$

An example of this formula in our scenario would be the following rule, which implies that users of similar age rate the same book similarly:

$$0.25\ \texttt{hasRating}(user_1, book, r) \wedge \texttt{shareAge}(user_1, user_2) \wedge \texttt{sameRating}(r, r) \Rightarrow \texttt{rates}(user_2, book, r)$$

User-preference Dependency: These rules consider only the dependency on the similarity of users preference behavior with respect to the relation that we want to predict.

Formula F.4

$$w_4\ \texttt{hasRelation}_j(u_1, o_1, r_1) \wedge \texttt{hasRelation}_j(u_2, o_1, r_1) \wedge \texttt{sameFeature}_k(r_1, r_1) \wedge \texttt{sameFeature}_k(r_2, r_2) \wedge \texttt{hasRelation}_j(u_1, o_2, r_2) \Rightarrow \texttt{query\_relation}_j(u_2, o_2, r_2)$$

For example:

$$0.78\ \texttt{hasRating}(u_1, book_1, r_1) \wedge \texttt{hasRating}(u_2, book_1, r_1) \wedge \texttt{sameRating}(r_1, r_1) \wedge \texttt{sameRating}(r_2, r_2) \wedge \texttt{hasRating}(u_1, book_2, r_2) \Rightarrow \texttt{rates}(u_2, book_2, r_2)$$

(a)Predicate Schema	(b) Evidence
*Rating(rating)	*Rating(1)
*User(person)	*Rating(2)
*Book(obj)	*Rating(3)
*Author(pers)	*User("Sara")
*Rating(rating)	*Book("book1")
*Country(cntr)	hasCountry("Sara","denmark")
*hasAge(person, age)	hasAge("Sara",32)
*hasCountry(pers, cntr)	hasRating("Sara", "book1", 5)
*hasAuthor(obj, author)	
*hasRating(pers, obj, rating)	sameUser("Sara","Sara")
*sameUser(pers, person)	*User("Bob")
*sameCountry(cntr, cntr)	hasAge("Bob",30)
*sameBook(obj, obj)	hasCountry("Bob","denmark")
*sameAuthor(pers, pers)	sameUser("Bob","Bob")
*sameRating(rating, rating)	shareCountry("Sara", "Bob")
shareCountry(pers, pers)	shareAge("Sara", "Bob")
shareAuthor(obj, obj)	...
shareAge(pers, pers)	
(c) Query	rates("Bob", "book1", r)

TABLE I. A SAMPLE OF THE PROPOSED HMLN MODEL

C. Probabilistic Inference

Our task is to predict the probability of the query predicates, such as $rates("Bob","book1",r)$, given the evidence in the system. As such, we perform marginal inference in order to estimate the marginal probability of the atoms that compose our queries. The inference process consists of two main steps: grounding and then searching.

1) User Network Selection: When generating the MLNs for large datasets, if we include all the users with whom a particular user u_i shares a relation (e.g. all the users who share same ratings with u_i), then the networks can become very large and inference is easily intractable. As such, we propose the following approach for network grounding in order to achieve better scalability during inference.

A network (the set of evidence, rules, and query formulae) is independently constructed for each user. This is what we referr to as *user graph* G_i, which is centered around a particular user u_i. Each *user graph* G_i contains information about the user u_i, and the objects to which she has direct relationships (e.g. books rated). We also include in the graph a set K of "neighboring" users and their respective profile information, as well as the objects related to them.

The process of neighbor selection consists in filtering those users that share relations with u_i (e.g. rated same books, or have the same age/country, etc.). Afterwards, the users are

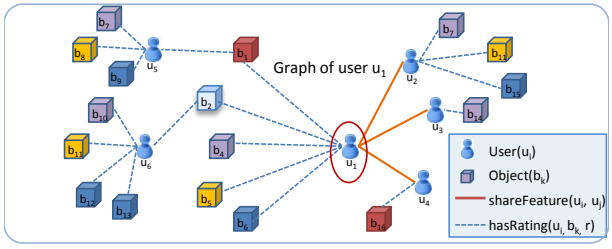

Fig. 2. Individual *User Graph*

ordered by a quantifiable measure of the relationship value they share with u_i, (e.g. number of co-rated books), and the top-K set of neighbors is finally selected. The last step is to include in the graph G_i all the objects to which every neighboring user has direct relations (i.e. books rated). An illustration of the user graph is depicted in Figure 2.

Note that grounding is now performed for each graph independently. For each user we predict the relationship value (e.g. rating) to a set of query objects. We have in the end the set of all user graphs $G_1, G_i, ..., G_{|U|}$.

Grounding. For each user graph G, we fix its hMLN-based schema σ and domain of known constants C. Given the hMNL-based set of formula $\bar{F} = \{F_1, ..., F_N\}$ of G (in *clausal form*) with weights $w_1, ..., w_N$, they define a probability distribution over *possible worlds*.

To construct this probability distribution, the first step is **grounding**: given a formula F with free variables $\bar{x} = (x_1, ..., x_m)$, for each constant $\bar{c} \in C^m$ we create a new formula $g_{\bar{c}}$, called a *ground clause*, which denotes the result of substituting each variable x_i of F with c_i. This process is performed for each formula F_i (for $i = 1...N$), where each ground clause g of F_i is assigned the same weight w_i. The set of obtained ground clauses of $\bar{F}$ corresponds to a hypergraph where each atom is a node and each clause is a hyperedge. This graph structure is a Markov network (Sec. III), also referred to as Markov Random Field (MRF).

In a Markov network, for any possible world (instance) I, a ground clause g is *violated* if $w(g) > 0$ and g is false in I, or if $w(g) \leq 0$ and g is true in I. We denote the set of ground clauses violated in a world I as $V(I)$. The cost of the world I is $cost(I) = \sum_{g \in V(I)} |w(g)|$. A lowest cost world I is called a *most likely world*. In order to find the most likely world or estimate the marginal probabilities of its atoms, we need to perform inference over the grounded network for each user graph.

Marginal Inference. In our approach, we are interested to compute the highest probabilities for the queries posed as part of the relation prediction task. This consists in estimating the marginal probability of the query atoms, which is the process of marginal inference.

Inference in MLNs is often regarded as infeasible because of the scalability issues associated with them. Yet, current state-of-the-art implementations show remarkable progress in overcoming these restrictions. We deploy marginal inference based on the MC-SAT algorithm [10], which applies slice sampling to Markov logic in combination with satisfiability testing by calling a heuristic SAT sampler. We apply the inference algorithm as implemented in the MLN inference engine Tuffy [2], which is recently shown to outperform all other engines in quality and efficiency [9].

D. Weight Learning

In our approach, we learn the weights of the formula discriminatively (maximizing the conditional likelihood of the query predicates given the evidence ones). Weight learning takes as input a training dataset and an MLN program without weights, then tries to compute the optimal weights of the MLN rules by maximizing the likelihood of the training data.

We use Diagonal Newton discriminative learner [6] as implemented in Tuffy. In our approach, we learn the weights for each user network separately, then use their mean for the formulae that compose our final set.

V. Experimental Evaluation

Evaluation methods for recommender systems are manifold, comprising statistical techniques to measure deviations of predicted and actual rating values, and approaches to estimate the utility of the recommendation list for the active user, e.g., precision and recall known from information retrieval.

In order to provide a comprehensive evaluation, we perform experiments for both aspects. As such, we organize the experiments in two parts: one for the *recommendations utility evaluation*, and the other for the *error deviation evaluation*. We chose to conduct experiments in three different datasets, in order to show not only the feasibility, but also the empirical expressiveness of our model.

A. Datasets

For the evaluation of error deviation, the experiments are conducted on two publicly available datasets:

MovieLens[3]: the original MovieLens dataset contains 10 million ratings (1-5 scales) from 71576 users and 10681 movies. For a better comparison with existing approaches, we follow the evaluation procedure of Shi et al. [14], by selecting a subset with the first 5000 users and 5000 movies according to the identifiers in the original dataset. In the following, this dataset is denoted as ML.

LibraryThing[4]: the original LibraryThing dataset contains ca. 750 thousand ratings from 7279 users and 37232 books, and in the subset we also select the first 5000 users and 5000 books. This dataset is denoted as LT.

As in [14], we chose the subset selection procedure rather than random selection, in order to ensure accurate performance comparison and future experimental reproducibility. The statistics of all the datasets are summarized in Table II.

TABLE II. Statistics of the datasets ML and LT

	Nr. users	Nr. items	Nr. Ratings	Sparseness
ML	5000	5000	584628	97.70%
LT	5000	5000	179419	99.30%

For the utility evaluation case, we perform another set of experiments on the following publicly available dataset from the recent initiatives on information heterogeneity in recommender systems [2]:

BookCrossing[5] is an online book club where users can rate books . In prior work [17], book ratings were collected from this site.[6] We performed a cleanup of the data, since it is quite noisy: there are invalid ISBNs, and some of the ISBNs in the rating file cannot be found in the book description file. Statistics of this dataset, denoted as BX, are displayed in Table III. We tests with various subsets by filtering users based on different numbers of minimal ratings.

[2] http://hazy.cs.wisc.edu/hazy/tuffy/

[3] http://www.grouplens.org/node/73

[4] http://ir.ii.uam.es/hetrec2011/datasets.html

[5] http://www.bookcrossing.com

[6] http://www.informatik.uni-freiburg.de/cziegler/BX/

TABLE III. STATISTICS OF THE BOOKCROSSING DATASET (BX)

Min. ratings	Nr. Users	Nr. Books	Nr. Ratings
5	5628	57,324	136,284
10	3056	52,528	119,563
30	1053	42,340	86,928
50	568	36,194	68,361

B. Experimental Setup

1) Experimental Protocol for Utility Case: We use decision-support metrics to evaluate the effectiveness of assisting users to select high-quality items from the overall set of items. We intend to judge how *relevant* a set of ranked recommendations is for the active user, thus, follow a methodology that estimates the utility of recommendations. As in Ma et al. [7], the first 50% of the ratings from each user are utilized for training, and the rest are utilized for testing.

At first, we select the users with at least 5 ratings ($min\_ratings = 5$). In addition, we perform other tests applying different values of $min\_ratings$, in order to see how the approach reacts to the cases when the users provide more preference judgments. We generate top-10 recommendations lists and perform 10-fold cross-validation. For this analysis we use the BX dataset.

2) Experimental Protocol for Error Deviation Case: We follow the experimental procedure of Shi et al. [14], which even though focuses on the cross-domain recommendation task, offers extensive evaluations of a single domain case such as ours. The comparative analysis is performed on ML and LT datasets, where each is divided into training set (60%), test set (20%), and validation set (20%). For each user in the test set, a small set of ratings (denoted as UPL for user profile length) is held out and included in the training set. In our case, UPL is set to 5. The rest of the ratings is used for testing applying 10-fold cross-validation.

3) Evaluation Metrics: For the utility evaluation case, we use established metrics of precision and recall. The recall metric [13] finds the percentage of test set objects in the dataset T_o occurring in recommendation list R_o, with respect to the overall number of test set objects $|T_o|$:

$$Recall = 100 \cdot \frac{|T_o \cap R_o|}{|T_o|} \quad (1)$$

Precision represents the percentage of test set objects occurring in the recommendation list, with respect to the size of the recommendation list:

$$Precision = 100 \cdot \frac{|T_o \cap R_o|}{|R_o|} \quad (2)$$

In addition, we report on the F1 measure as a combined metric for precision and recall.

For the evaluation case of measuring error deviations, we use mean absolute error (MAE) as the standard evaluation metric [7], [14] for measuring recommendation performance on rating-based recommender domains:

$$MAE = \frac{\sum |r_{u,o} - \bar{r}_{u,o}|}{|T_o|} \quad (3)$$

where $r_{u,o}$ denotes the rating that user u gave to object o, and $\bar{r}_{u,o}$ denotes the rating that user u gave to object o which ispredicted by our approach, and $|T_o|$ denotes the number of tested ratings.

4) Parameter Setting: For weight learning, we use the following parameters: number of samples for MC-SAT is set to 50, and max. iterations to 100. For user network grounding, we set the neighbor clustering parameter(Sec. IV-C1) to $K = 20$.

C. Performance Comparison

We compare the performance of the proposed hMLN with a set of alternative recommendation approaches listed below:

- **User-based Collaborative Filtering (UBCF)** is a representative of memory-based CF approaches, being one of the most popular recommendation techniques, because of its simplicity and high quality of recommendations. We apply Pearson correlation for similarity values and set neighborhood to 50.
- **Item-based Collaborative Filtering (IBCF)** is chosen as another popular recommendation method. It is a representative of model-based CF methods [13], which in addition to CF considers the item-item similarities.
- **Probabilistic matrix factorization (PMF)** [12] is a state-of-the-art model-based CF approach. The regularization parameter is set to 0.01.

We summarize below the results of the comparative analysis and the observations regarding the hMNL approach.

TABLE IV. RECOMMENDATION PERFORMANCE ON DATASET BX

Metrics	Min.Ratings=5		
	UBCF	IBCF	hMLN
Recall	5.76	**7.32**	5.12
Precision	3.69	3.64	**4.67**
F1	4.49	4.86	**4.88**

1) Results of Utility Evaluation: Results illustrated in Table IV show that user-based CF and item-based CF exhibit almost the same accuracy, indicated by the precision values. Their difference in recall shows a behavior change with respect to the types of users used in the scoring. Our approach, hMNL, yields a recall value similar to the user-based method, but outperformes the other methods in precision. hMLN also gives the best F1 value. This can be explained by the fact that our method is accurate in making predictions, but the restrictions we have put in the size of networks per user cause a decrease in the number of relevant items predicted. It means that our choice on achieving scalability comes as a trade-off with recall.

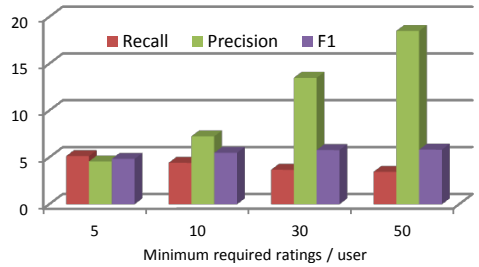

Fig. 3. hMLN performance for varying $min\_ratings$

In Fig.3, we see that for increasing values of the minimum ratings applied to the evaluation setting, hMLN provides much

higher accuracy of the recommendation results. In particular, precision largely increases when we filter users that have 30-50 ratings. From the observations, we draw a conclusion that hMLN is precise in making relevant recommendations, but its coverage is restricted to the scale of networks that we construct for inferencing. With larger networks recall can increase, but scalability needs to be always taken into consideration to ensure tractable solutions.

2) Results of Error Deviations: Since IBCF is computationally expensive, we have used UBCF and PMF as better representative of state-of-the-art for this comparative analysis.

TABLE V. RECOMMENDATION PERFORMANCE IN THE ERROR DEVIATION CASE BETWEEN HMLN AND BASELINE APPROACHES

Metrics	UBCF	PMF	hMLN
ML(MAE)	0.833	0.831	**0.641**
LT(MAE)	0.857	0.771	**0.738**

As can be seen in Table V, hMLN outperforms the other approaches with regard to the mean absolute error in both datasets ML and LT. Our approach ensures high accuracy because of the power to handle the probability distributions of the closest dependencies that help in defining users' ratings.

VI. CONCLUSIONS

In this work, we present a generic model for generating recommendations following a hybrid approach. The model is based on Markov logic, which is a simple and powerful language that combines first-order logic and probabilistic graphical models. We experimentally verify the efficacy of our theoretical model, and demonstrate that our method achieves better accuracy than other state-of-the-art recommendation approaches. Experiments also show that precision largely increases when considering users with more explicit preference judgments (even when this number is not very high).

This work opens an avenue for future research, for example, investigating how the scale of user graphs proposed for MLN construction influence the recommendation performance (w.r.t. recall). An interesting direction is to address methods from structure learning for such solution. We also plan to show the extent of the applicability of our model in various domains.

REFERENCES

[1] D. Agarwal and B.-C. Chen. Regression-based latent factor models. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 19–28, New York, NY, USA, 2009. ACM.

[2] I. Cantador, P. Brusilovsky, and T. Kuflik. 2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In *Proceedings of the 5th ACM conference on Recommender systems*, RecSys 2011, New York, NY, USA, 2011. ACM.

[3] L. Getoor and M. Sahami. Using probabilistic relational models for collaborative filtering. In *In Workshop on Web Usage Analysis and User Profiling (WEBKDD'99*, 1999.

[4] L. Getoor and B. Taskar. *Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2007.

[5] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, Aug. 2009.

[6] D. Lowd and P. Domingos. Efficient weight learning for markov logic networks. In *Proceedings of the 11th European conference on Principles and Practice of Knowledge Discovery in Databases*, PKDD 2007, pages 200–211, Berlin, Heidelberg, 2007. Springer-Verlag.

[7] H. Ma, I. King, and M. R. Lyu. Effective missing data prediction for collaborative filtering. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 39–46, New York, NY, USA, 2007. ACM.

[8] R. J. Mooney and L. Roy. Content-based book recommending using learning for text categorization. In *Proceedings of the fifth ACM conference on Digital libraries*, DL '00, pages 195–204, New York, NY, USA, 2000. ACM.

[9] F. Niu, C. Ré, A. Doan, and J. Shavlik. Tuffy: scaling up statistical inference in markov logic networks using an rdbms. *Proc. VLDB Endow.*, 4(6):373–384, Mar. 2011.

[10] H. Poon and P. Domingos. Sound and efficient inference with probabilistic and deterministic dependencies. In *Proceedings of the 21st national conference on Artificial intelligence - Volume 1*, AAAI'06, pages 458–463. AAAI Press, 2006.

[11] M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62:107–136, 2006.

[12] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *NIPS*. Curran Associates, Inc., 2007.

[13] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl. Application of dimensionality reduction in recommender system – a case study. In *WebKDD Workshop*, 2000.

[14] Y. Shi, M. Larson, and A. Hanjalic. Generalized tag-induced cross-domain collaborative filtering. *CoRR*, abs/1302.4888, 2013.

[15] Z. Xu, V. Tresp, A. Rettinger, and K. Kersting. Social network mining with nonparametric relational models. In *Proceedings of the Second international conference on Advances in social network mining and analysis*, SNAKDD'08, pages 77–96, Berlin, Heidelberg, 2010. Springer-Verlag.

[16] Y. Zhang and J. Koren. Efficient bayesian hierarchical user modeling for recommendation system. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 47–54, New York, NY, USA, 2007. ACM.

[17] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen. Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web*, WWW '05, pages 22–32, New York, NY, USA, 2005. ACM.