Learning in Probabilistic Graphs exploiting Language-Constrained Patterns

Claudio Taranto, Nicola Di Mauro, and Floriana Esposito

Department of Computer Science, University of Bari "Aldo Moro"
via E. Orabona, 4 - 70125, Bari, Italy
{claudio.taranto,ndm,esposito}@di.uniba.it

Abstract. The probabilistic graphs framework models the uncertainty inherent in real-world domains by means of probabilistic edges whose value quantifies the likelihood of the edge existence or the strength of the link it represents. The goal of this paper is to provide a learning method to compute the most likely relationship between two nodes in a framework based on probabilistic graphs. In particular, given a probabilistic graph we adopted the language-constrained reachability method to compute the probability of possible interconnections that may exists between two nodes. Each of these connections may be viewed as feature, or a factor, between the two nodes and the corresponding probability as its weight. Each observed link is considered as a positive instance for its corresponding link label. Given the training set of observed links a L2-regularized Logistic Regression has been adopted to learn a model able to predict unobserved link labels.

1 Introduction

Over the last few years the extension of graph structures with uncertainty has become an important research topic [1–4], leading to *probabilistic graph* model. Probabilistic graphs model uncertainty by means of probabilistic edges whose value quantifies the likelihood of the edge existence or the strength of the link it represents. One of the main issues in probabilistic graphs is how to compute the connectivity of the network. The network reliability problem [5] is a generalization of the pairwise reachability, in which the goal is to determine the probability that all pairs of nodes are reachable from one another. Unlike a deterministic graph in which the reachability function is a binary value function indicating whether or not there is a path connecting two nodes, in the case of probabilistic graphs the function assumes probabilistic values.

The concept of *reachability* in probabilistic graphs is used, along with its specialization, as a tool to compute how two nodes in the graph are likely to be connected. Reachability plays an important role in wide range of applications, such as in peer-to-peer networks, for probabilistic-routing problem, in road network, and in trust analysis in social networks. Reachability is quite similar to the general concept of *link prediction* [6], whose task may be formalized as follows.

Given a networked structure (V, E) made up of a set of data instances V and set of observed links E among some nodes in V, the task corresponds to predict how likely should exist an unobserved link between two nodes. The extension to probabilistic graphs adds an important ingredient that should be adequately exploited. The key difference with respect to classical link prediction is that here the observed connections between two nodes cannot be considered always true, and hence methods exploiting probabilistic links are needed.

The goal of this paper is to provide a learning method to compute the most likely relationship between two nodes in probabilistic graphs. In particular, given a probabilistic graph we adopted the reachability tool to compute the probability of some possible interconnections that may exists between two nodes. Each of these connections may be viewed as a feature, or a pattern, between the two nodes and the corresponding probability as its weight. Each observed labeled link is considered as a positive instance for its corresponding link label. The link label corresponds to the value of the output variable y_i, and the features between the two nodes, computed with the reachability tool, correspond to the components of the corresponding vector $\mathbf{x}_i$. Given the training set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, obtained from n observed links, a L2-regularized Logistic Regression has been adopted to learn a model to be used to predict unobserved link labels.

The proposed approach is quite similar to that of *propositionalization* proposed in the fields of Statistical Relational Learning [7], where the relational data are flattened to a propositional representations using relational fetures in order to have efficient learning results. Here the further problem that we have to handle is that the relational representation is uncertain.

The application domain we chosen corresponds to the problem of recommender systems [8], where the aim is to predict the unknown rating between an user and an item. The experiments on a real-world dataset prove that the proposed approach achieves better results than that obtained with models induced by Singular Value Decomposition (SVD) [9] on the user-item ratings matrix, representing one of the best recent methods for this kind of task [10].

2 Probabilistic Graphs

Let $G = (V, E)$, be a graph where V is a collection of nodes and $E \in V \times V$ is the set of edges, or relationships, between the nodes.

Definition 1 (Probabilistic graph). *A* probabilistic graph *is a system* $G = (V, E, \Sigma, l_V, l_E, s, t, p_e)$*, where* (V, E) *is an directed graph,* V *is the set of nodes,* E *is the set of ordered pairs of nodes where e=(s,t),* Σ *is a set of labels,* $l_V : V \rightarrow \Sigma$ *is a function assigning labels to nodes,* $l_E : E \rightarrow \Sigma$ *is a function assigning labels to the edges,* $s : E \rightarrow V$ *is the source node of an edge,*

$t : E \to V$ *is the target node of an edge,* $p_e : E \to [0, 1]$ *is a function assigning* existence probability *values to the edges.*

The existence probability $p_e(a)$ of an edge $a = (u, v) \in E$ is the probability that the edge a, between u and v, can exist in the graph. A particular case of probabilistic graph is the *discrete graph*[1], where binary edges between nodes represent the presence or absence of a relationship between them, i.e., the existence probability value on all observed edges is 1. The *possible world semantics* is usually used for probabilistic graphs. We can imagine a probabilistic graph G as a sampler of worlds, where each world is an instance of G. A discrete graph G' is sampled from G according to the probability distribution P_e, denoted as $G' \sqsubseteq G$, when each edge $a \in E$ is selected to be an edge of G' with probability $p_e(a)$. Edges labeled with probabilities are treated as mutually independent random variables indicating whether or not the corresponding edge belongs to a discrete graph.

Assuming independence among edges, the probability distribution over discrete graphs $G' = (V, E') \sqsubseteq G = (V, E)$ is given by

$$P(G'|G) = \prod_{a \in E'} p_e(a) \prod_{a \in E \setminus E'} (1 - p_e(a)). \tag{1}$$

Definition 2 (Simple path). *Given an uncertain graph* G*, a* simple path *of a length* k *from* u *to* v *in* G *is an acyclic path denoted as a sequence of edges* $p_{u,v} = \langle e_1, e_2, \dots e_k \rangle$*, such that* $e_1 = (u, v_1)$*,* $e_k = (v_{k_1}, v)$*, and* $e_i = (v_{i-1}, v_i)$ *for* $1 < i < k$*.*

Given an uncertain graph G, and $p_{u,v}$ a path in G from node u to node v, $\ell(p_{u,v}) = l(e_1)l(e_2) \cdots l(e_k)$ denotes the concatenation of labels of all the edges in $p_{u,v}$. We adopt a *regular expression* $\mathtt{R}$ to denote what is the exact sequence of labels that the path must contain.

Definition 3 (Language-constrained simple path). *Given a probabilistic graph* G *and a* regular expression $\mathtt{R}$*, a* language constrained simple path *is a simple path* p *such that* $\ell(p) \in L(\mathtt{R})$*.*

2.1 Inference

Given a probabilistic graph G a main task corresponds to compute the probability that there exists a simple path between two nodes u and v, that is, querying for the probability that a randomly sampled discrete graph contains a simple path between u and v. More formally, the *existence probability* $P_e(q|G)$ of a

[1] Sometimes called *certain graph*.

simple path q in a probabilistic graph G corresponds to the marginal $P((q, G')|G)$ with respect to q:

$$P_e(q|G) = \sum_{G' \sqsubseteq G} P(q|G') \cdot P(G'|G) \tag{2}$$

where $P(q|G') = 1$ if there exits the simple path q in G', and $P(q|G') = 0$ otherwise. In other words, the existence probability of the simple path q is the probability that the simple path q exists in a randomly sampled discrete graph.

Definition 4 (Language-constrained simple path probability). *Given a probabilistic graph G and a* regular expression $\mathtt{R}$*, the* language-constrained simple path probability *of $L(\mathtt{R})$ is*

$$P_e(q|L(\mathtt{R}), G) = \sum_{G' \sqsubseteq G} P(q|G', L(\mathtt{R})) \cdot P(G'|G) \tag{3}$$

where $P(q|G', L(\mathtt{R}) = 1$ if there exists a simple path q in G' such that $\ell(q) \in L(\mathtt{R})$, and $P(q|G', L(\mathtt{R})) = 0$ otherwise.

The previous definition give us the possibility to compute the probability of a set of simple path queries, or patterns, fulfilling the structure imposed by a regular expression. In this way we are interested in discrete graphs that contain at least one simple path belonging to the language denoted by the regular expression.

Computing the existence probability directly using (2) or (3) is intensive and intractable for large graphs since the number of discrete graphs to be checked is exponential in the number of probabilistic edges. It involves computing the existence of the simple path in every discrete graph and accumulating their probability. A natural way to overcome the intractability of computing the existence probability of a simple path is to approximate it using a Monte Carlo sampling approach [11]: 1) we sample n possible discrete graphs, $G_1, G_2, \ldots G_n$ from G by sampling edges uniformly at random according to their edge probabilities; and 2) we check if the simple path exists in each sampled graph G_i. This process provides the following basic sampling estimator for $P_e(q|G)$:

$$P_e(q|G) \approx \widehat{P_e(q|G)} = \frac{\sum_{i=1}^{n} P(q|G')}{n} \tag{4}$$

Note that is not necessary to sample all edges to check whether the graph contains the path. For instance, assuming to use an iterative depth first search (DFS) procedure to check the path existence. When a node is just visited, we will sample all its adjacent edges and pushing them into the stack used by the iterative procedure. We will stop the procedure either when the target node is reached or when the stack is empty (non existence).

3 Link Classification

After having defined the probabilistic graph, now we can adopt language-constrained simple paths in order to extract probabilistic features (patterns) to describe the link between two nodes in the graph.

Given a probabilistic graph G, with the set V of nodes and the set E of edges, and $Y \subseteq \Sigma$ a set of edge labels, we have a set of edges $D \subseteq E$ such that for each element $e \in D$: $l_E(e) \in Y$. In particular D represents the set of observed links whose label belongs to the set Y. Given the set of training links D and the set of labels Y we want to learn a model able to correctly classify unobserved links. A way to solve the classification task can be that of using a language based classification approach. Given an unobserved edge $e_i = (u_i, v_i)$, in order to predict its class $\widehat{y_i} \in Y$ we can solve the following maximization problem:

$$\widehat{y_i} = \arg\max_j P(q_j(u_i, v_i)|G), \tag{5}$$

where $q_j(u_i, v_i)$ is the unknown link with label $q_j \in Y$ between the nodes u_i and v_i. In particular, the maximization problem corresponds to compute the link prediction for each $q_j \in Y$ and then choosing that label with maximum likelihood. The previous link prediction task is based on querying the probability of some language-constrained simple path. In particular, predicting the probability of the label q_j as $P(q_j(u_i, v_i)|G)$ in (5) corresponds to compute the probability $P(q|G)$ for a query path in a language L_j, i.e., computing $P(L_j|G)$ as in (3):

$$\widehat{y_j} = \arg\max_j P(q_j(u_i, v_i)|G) \approx \arg\max_j P(q|L_j, G). \tag{6}$$

The previous query based approach consider the languages used to compute the (6) as independent form each other without considering any correlation between them. A more interesting approach that we want investigate in this paper is to learn from the probabilistic graph a linear model of classification combining the prediction of each language constrained simple path. In particular, given an edge e and a set of k languages $\mathcal{L} = \{L_1, \ldots, L_k\}$, we can generate k real valued features x_i where $x_i = P(q|L_i, G)$, $1 \leq i \leq k$. The original training set of observed links D can hence be transformed into the set of instances $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1,\ldots,n}$, where $\mathbf{x}_i$ is a k-component vector of features $x_{ij} \in [0, 1]$, and y_i is the class label of the corresponding example $\mathbf{x}_i$.

Linear classification represents one of the most promising learning technique for problems with a huge number of instances and features aiming at learning a weight vector $\mathbf{w}$ as a model. L2-regularized Logistic Regression belongs to the class of linear classifier and solves the following unconstrained

optimization problem:

$$\min_{\mathbf{w}} f(\mathbf{w}) = \left(\frac{\mathbf{w}^T\mathbf{w}}{2} + C\sum_{i=1}^{n} \log(1+\exp(-y_i\mathbf{w}^T\mathbf{x}_i)) \right), \tag{7}$$

where $\log(1+\exp(-y_i\mathbf{w}^T\mathbf{x}_i)) = \xi(\mathbf{w};\mathbf{x}_i,y_i)$ denotes the specific loss function, $\frac{1}{2}\mathbf{w}^T\mathbf{w}$ is the regularized term, and $C > 0$ is a penalty parameter. The decision function corresponds to $\text{sgn}(\mathbf{w}^t\mathbf{x}_i)$. In case of binary classification $y_i \in \{-1,+1\}$, while for multi class problems the one vs the rest strategy can be used. Among many methods for training logistic regression models, such as iterative scaling, nonlinear conjugate gradient, quasi Newton, a new efficient and robust truncated Newton, called trust region Newton method, has been proposed [12]. In order to find the parameters $\mathbf{w}$ minimizing $f(\mathbf{w})$ it is necessary to set the derivative of $f(\mathbf{w})$ to zero. Denoting with $\sigma(y_i\mathbf{w}^T\mathbf{x}_i) = (1+exp(-y_i\mathbf{w}^T\mathbf{x}_i))^{-1}$, we have:

$$\frac{\partial f(\mathbf{w})}{\partial \mathbf{w}} = \mathbf{w} + C\sum_{i=1}^{n} \left(\sigma(y_i\mathbf{w}^T\mathbf{x}_i) - 1\right) y_i\mathbf{x}_i = 0.$$

To solve the previous score equation, the Newton method requires the Hessian matrix:

$$\frac{\partial^2 f(\mathbf{w})}{\partial \mathbf{w}\partial \mathbf{w}^T} = \mathbf{I} + C\mathbf{X}^T\mathbf{D}\mathbf{X},$$

where $\mathbf{X}$ is the matrix of the $\mathbf{x}_i$ values, $\mathbf{D}$ is a diagonal matrix of weights with ith diagonal element $\sigma(y_i\mathbf{w}^T\mathbf{x}_i)(1-\sigma(y_i\mathbf{w}^T\mathbf{x}_i))$, and $\mathbf{I}$ is the identity matrix.

The Newton step is $\mathbf{w}^{\text{new}} \leftarrow \mathbf{w}^{\text{old}} + \mathbf{s}^{\text{old}}$, where $\mathbf{s}^{\text{old}}$ is the solution of the following linear system:

$$\frac{\partial^2 f(\mathbf{w}^{\text{old}})}{\partial \mathbf{w}\partial \mathbf{w}^T}\mathbf{s}^{\text{old}} = -\frac{\partial f(\mathbf{w}^{\text{old}})}{\partial \mathbf{w}}.$$

Instead of using this update rule, [12] propose a robust and efficient trust region Newton method, using new rules for updating the trust region, whose corresponding algorithm has been implemented in the LIBLINEAR[2] system.

4 EXPERIMENTAL EVALUATION

The application domain we chosen to validate the proposed approach is that of recommender systems. In some domains both data and probabilistic relationships between them are observable, while in other domain, like in this used in this paper, it is necessary to elicit the uncertain relationships among the given evidence.

[2] http://www.csie.ntu.edu.tw/~cjlin/liblinear.

4.1 Probabilistic graph creation

A common approach to elicit probabilistic hidden relationships between data is based on using similarity measures. To model the data with a graph we can adopt different similarity measures for each type of node involved in the relationships.

In a recommender system we have two types of entities: the users and the items, and the only observed relationship corresponds to the ratings that a user has assigned to a set of items. The goal is to predict the rating a user could assign to an object that he never rated in the past. In the collaborative filtering approach there are two methods to predict unknown rating exploiting users or items similarity. User-oriented methods estimate unknown ratings based on previous ratings of similar users, while in item-oriented approaches ratings are estimated using previous ratings given by the same user on similar items.

Let U be a set of n users and I a set of m items. A rating r_{ui} indicates the preference degree the user u expressed for the item i, where high values mean stronger preference. Let S_u be the set of items rated from user u. A user-based approach predicts an unobserved rating $\widehat{r_{ui}}$ as follows:

$$\widehat{r_{ui}} = \overline{r_u} + \frac{\sum_{v \in U | i \in S_u} \sigma_u(u,v) \cdot (r_{vi} - \overline{r_v})}{\sum_{v \in U | i \in S_u} |\sigma_u(u,v)|} \tag{8}$$

where $\overline{r_u}$ represents the mean rating of user u, and $\sigma_u(u,v)$ stands for the similarity between users u and v, computed, for instance, using the Pearson correlation: $\sigma_u(u,v) = \frac{\sum_{a \in S_u \cap S_v} (r_{ua} - \overline{r_u}) \cdot (r_{va} - \overline{r_v})}{\sqrt{\sum_{a \in S_u \cap S_v} (r_{ua} - \overline{r_u})^2 \sum_{a \in S_u \cap S_v} (r_{va} - \overline{r_v})^2}}$.

On the other side, item-based approaches predict the rating of a given item using the following formula:

$$\widehat{r_{ui}} = \frac{\sum_{j \in S_u | j \neq i} \sigma_i(i,j) \cdot r_{uj}}{\sum_{j \in S_u | j \neq i} |\sigma_i(i,j)|}, \tag{9}$$

where $\sigma_i(i,j)$ is the similarity between the item i and j.

These neighbourhood approaches see each user connected to other users or consider each item related to other items as in a network structure. In particular they rely on the direct connections among the entities involved in the domain. However, as recently proved, techniques able to consider complex relationships among the entities, leveraging the information already present in the network, involves an improvement in the processes of querying and mining [13, 14].

Recommender Probabilistic Graph Given the set of observed ratings $\mathcal{K} = \{(u,i,r_{ui}) | r_{ui} \text{ is known}\}$, we add a node with label `user` for each user in $\mathcal{K}$, and a node with label `item` for each item in $\mathcal{K}$. The next step is to add

the edges among the nodes. Each edge is characterized by a label and a probability value, which should indicate the degree of similarity between the two nodes. Two kind of connections between nodes are added. For each user u, we added an edge, labeled as $\texttt{simU}$, between u and the k most similar users to u. The similarity between two users u and v is computed adopting a weighted Pearson correlation between the items rated by both u and v. In particular, the probability of the edge $\texttt{simU}$ connecting two users u and v is computed as: $P(\texttt{simU}(u,v)) = \sigma_u(u,v) \cdot w_u(u,v)$, where $\sigma_u(u,v)$ is the Pearson correlation between the vectors of ratings corresponding to the set of items rated by both user u and user v, and $w_u(u,v) = |S_u \cap S_v|/|S_u \cup S_v|$.

For each item i, we added an edge, with label $\texttt{simI}$, between i and the most k similar items to i. In particular, the probability of the edge $\texttt{simI}$ connecting the item i to the item j has been computed as: $P(\texttt{simI}(i,j)) = \sigma_i(i,j) \cdot w_i(i,j)$, where $\sigma_i(i,j)$ is the Pearson correlation between the vectors corresponding to the histogram of the set of ratings for the item i and the item j, and $w_i(i,j) = |\overline{S}_i \cap \overline{S}_j|/|\overline{S}_i \cup \overline{S}_j|$, where $\overline{S}_i$ is the set of users rating the item i.

Finally, edges with probability equal to 1, and with label $\texttt{r}_k$ between the user u and the item i, denoting the user u has rated the item i with a score equal to k, are added for each element (u, i, r_k) belonging to $\mathcal{K}$.

4.2 Feature construction

Let us assume that the values of r_{ui} are discrete and belonging to a set R. Given the recommender probabilistic graph G, the query based classification approach try to solve the problem $\widehat{r_{ui}} = \arg\max_j P(\texttt{r}_j(u,i)|G)$, where $\texttt{r}_j(u,i)$ is the unknown link with label $\texttt{r}_j$ between the user u and the item i. This link prediction task is based on querying the probability of some language constrained simple path. For instance, a user-based collaborative filtering approach may be obtained by querying the probability of the edges, starting from a user node and ending to an item node, denoted by the regular expression $L_i = \{\texttt{simU}^1\texttt{r}_i^1\}$. In particular, predicting the probability of the rating j as $P(\texttt{r}_j(u,i))$ corresponds to compute the probability $P(q|G)$ for a query path in L_j, i.e., $\widehat{r_{ui}} = \arg\max_j P(\texttt{r}_j(u,i)|G) \approx \arg\max_j P(L_j|G)$. In the same way, item-based approach could be obtained by computing the probability of the paths constrained by the language $L_i = \{\texttt{r}_i^1\texttt{simI}^1\}$.

The power of the proposed framework gives us the possibility to construct more complex queries such as that constrained by the language $L_i = \{\texttt{r}_i\texttt{simI}^n : 1 \leq n \leq 2\}$, that gives us the possibility to explore the graph by considering not only direct connections. Hybrid queries, such as those constrained by the language $L_i = \{\texttt{r}_i\texttt{simI}^n : 1 \leq n \leq 2\} \cup \{\texttt{simU}^m\texttt{r}_i^1 : 1 \leq m \leq 2\}$, give us the possibility to combine the user information with item information.

In order to use the feature based classification approach proposed in this paper we can define a set of regular expression $\mathcal{L}$ and then computing for each language $L_i \in \mathcal{L}$ the probability $P(L_i|G)$ between a given user and all the items the user rated. In particular, the set of observed ratings $\mathcal{K} = \{(u, i, r_{ui})|r_{ui}$ is known$\}$ is mapped to the training set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1,\ldots,n}$, where x_{ij} is the probability $P(L_j|G)$ between the nodes u and i, and y_i is equal to r_{ui}.

The proposed link classification method has been implemented in the `Eagle` system [3] that provides a set of tools to deal with probabilistic graphs.

4.3 Dataset

In order to validate the proposed approach we used the MovieLens dataset[4], made available by the GroupLens research group at University of Minnesota for the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems. We used the MovieLens 100K version consisting of 100000 ratings (ranging from 1 to 5) regarding 943 users and 1682 movies. Each user has rated at least 20 movies and there are simple demographic info for the users (such as age, gender, occupation, and zip code).In this paper we used the ratings only without considering the demographic information. MovieLens 100K dataset is divided in 5 fold, where each fold present a training data (consisting of 80000 ratings) and a test data (with 20000 ratings).

For each training/testing fold the validation procedure followed the steps:

1. creating the probabilistic graph from the training ratings data set as reported in Section 4.1;
2. defining a set $\mathcal{L}$ of regular expressions to be used to construct a specific set of features as described in Section 4.2;
3. learning the L2-regularized Logistic Regression model; and
4. testing the ratings reported in the testing data set $\mathcal{T}$ by computing, for each pair $(u, i) \in \mathcal{T}$ the predicted rating adopting the learned classification model and comparing the result with the true prediction reported in $\mathcal{T}$.

For the graph construction, edges are added using the procedure presented in Section 4.1, where we set the parameter $n = 30$, indicating that an user or a film is connected, respectively, to 30 most similar users, resp. films. The value of each feature have been obtained with the Monte Carlo inference procedure by sampling M discrete graphs.

In order to construct the set of features, we proposed to query the paths belonging to the set of languages $\mathcal{L}$ reported in Table 1. The first language-constrained simple paths L_1 corresponds to adopt a user-based approach, while

[3] `http://www.di.uniba.it/~claudiotaranto/eagle.html`

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

the second language L_2 gives us the possibility to simulate an item-based approach. Then, we propose to extend the basic languages L_1 and L_2 in order to construct features that consider a neighbourhood with many nested levels. Finally, we constructed hybrid features by combining both the user-based and item-based methods and the large neighbourhood explored with paths whose length is greater than one (L_5, L_8 and L_9).

We defined two sets of features $\mathcal{F}_1 = \{L_1, L_2, L_3, L_4, L_5\}$, based on simple languages, and $\mathcal{F}_2 = \{L_3, L_4, L_5, L_6, L_7, L_8, L_9\}$, exploiting more complex queries. In order to learn the classification model as reported in Section 3, we used the L2-regularized Logistic Regression implementation included in the LIBLINEAR system [12].

Table 1. Language constrained simple paths used for the MovieLens dataset.

$L_1 = \{\texttt{simU}^1\texttt{r}_k^1\}$
$L_2 = \{\texttt{r}_k^1\texttt{simF}^1\}$
$L_3 = \{\texttt{r}_k^1\texttt{simF}^n : 1 \leq n \leq 2\}$
$L_4 = \{\texttt{simU}^n\texttt{r}_k^1 : 1 \leq n \leq 2\}$
$L_5 = \{\texttt{simU}^n\texttt{r}_k^1 : 1 \leq n \leq 2\} \cup \{\texttt{r}_k^1\texttt{simF}^n : 1 \leq n \leq 2\}$
$L_6 = \{\texttt{r}_k^1\texttt{simF}^n : 1 \leq n \leq 3\}$
$L_7 = \{\texttt{simU}^n\texttt{r}_k^1 : 1 \leq n \leq 3\}$
$L_8 = \{\texttt{simU}^n\texttt{r}_k^1 : 1 \leq n \leq 3\} \cup \{\texttt{r}_k^1\texttt{simF}^n : 1 \leq n \leq 3\}$
$L_9 = \{\texttt{simU}^n\texttt{r}_k^1 : 1 \leq n \leq 4\} \cup \{\texttt{r}_k^1\texttt{simF}^n : 1 \leq n \leq 4\}$

Given a set $\mathcal{T}$ of testing instances, the accuracy of the proposed framework has been evaluated according to the *macroaveraging mean absolute error* [15]: $MAE^M(\widehat{r_{ui}}, \mathcal{T}) = \frac{1}{k}\sum_{j=1}^{k}\frac{1}{|T_j|}\sum_{x_i \in T_j}|\widehat{r_{ui}} - r_{ui}|$, where $T_j \subset \mathcal{T}$ denotes the set of test rating whose true class is j.

4.4 Results

Table 2 shows the results obtained adopting the proposed approach implemented in the `Eagle` system when compared to those obtained with the RecSys SVD approach based implementation[5]. The first row reports the mean value of the MAE^M averaged on the five folds obtained with an SVD approach and with the proposed classification method as implemented in the `Eagle` system. As we can see the error achieved by our method is lower than that obtained by the SVD method. The results improve when we use the set $\mathcal{F}_2$ of features. The difference of the results obtained with the two methods is statistically significant, with a p-value for the t-test equal to 0.0000023 when using the set $\mathcal{F}_1$ of features, and

[5] `https://github.com/ocelma/python-recsys`

equal to 0.000000509 for the other set of features. The last two columns report the results of two baseline methods. The second last column reports the results obtained with a system that predicts a rating adopting a uniform distribution, while the last column reports the results of a system that uses a categorical distribution that predicts the value k of a rating with probability $p_k = |D_k|/N$, where D_k is the number of ratings belonging to the dataset having value k, and N is the total number of ratings.

Table 2. MAE^M values obtained with `Eagle` and SVD on MovieLens dataset.

Fold	`SVD`	`Eagle@`$\mathcal{F}_1$	`Eagle@`$\mathcal{F}_2$	`U`	`C`
1	0.9021	0.8424	0.8255		
2	0.9034	0.8332	0.8279		
3	0.9111	0.8464	0.8362		
4	0.9081	0.8527	0.8372		
5	0.9159	0.8596	0.8502		
Mean	0.908±0.006	0.847±0.01	0.835±0.01	1.6	1.51
p-value		2.3E-6	5.09E-7		

In Table 3 we can see the errors committed by each method for each rating. The rows for the methods `U` and `C` report the mean of the MAE^M value for each fold using a system adopting a uniform or a categorical distribution. The dataset is not balanced as and both the `SVD` and the proposed method adhere more to the categorical distribution proving that they are able to recognize the unbalanced distribution of the dataset

Table 3. MAE^M values for each class obtained with `Eagle` and SVD on MovieLens dataset.

Method	r1	r2	r3	r4	r5
`U`	2.0	1.4	1.2	1.4	2.0
`C`	2.53	1.65	1.00	0.89	1.47
`SVD`	1.62	1.03	0.55	0.44	0.88
`Eagle@`$\mathcal{F}_1$	1.16	0.76	0.67	0.60	1.02
`Eagle@`$\mathcal{F}_2$	1.11	0.75	0.68	0.62	1.00

5 CONCLUSIONS

In this paper we adopt the probabilistic graphs framework to deal with uncertain problems exploiting both edges probabilistic values and edges labels denoting the type of relationships between two nodes. In this paper the `Eagle` system

integrating a framework based on probabilistic graphs able to deal with link prediction problems adopting reachability has been presented. We proposed a learning method to compute the most likely relationship between two nodes in probabilistic graphs. Given the training set of observed links a L2-regularized Logistic Regression has been adopted to learn a model able to predict the label of unobserved links. The experimental evaluation proved that the proposed approach achieves better results when compared to that obtained with models induced by Singular Value Decomposition on the user-item ratings matrix, representing one of the best recent method for this kind of problem.

References

1. Potamias, M., Bonchi, F., Gionis, A., Kollios, G.: k-nearest neighbors in uncertain graphs. Proc. VLDB Endow. **3** (2010) 997–1008
2. Zou, Z., Gao, H., Li, J.: Discovering frequent subgraphs over uncertain graph databases under probabilistic semantics. In: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM (2010) 633–642
3. Zou, Z., Li, J., Gao, H., Zhang, S.: Finding top-k maximal cliques in an uncertain graph. International Conference on Data Engineering (2010) 649–652
4. III, J.J.P., Neville, J.: Methods to determine node centrality and clustering in graphs with uncertain structure. In: Proceedings of the Fifth International Conference on Weblogs and Social Media, The AAAI Press (2011)
5. Colbourn, C.J.: The Combinatorics of Network Reliability. Oxford University Press (1987)
6. Getoor, L., Diehl, C.P.: Link mining: a survey. SIGKDD Explorations **7**(2) (2005) 3–12
7. Getoor, L., Taskar, B.: Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning). The MIT Press (2007)
8. Desrosiers, C., Karypis, G.: A comprehensive survey of neighborhood-based recommendation methods. In Ricci, F., Rokach, L., Shapira, B., Kantor, P.B., eds.: Recommender Systems Handbook. Springer (2011) 107–144
9. Pryor, M.H.: The effects of singular value decomposition on collaborative filtering. Technical Report PCS-TR98-338, Dartmouth College, Computer Science, Hanover, NH (1998)
10. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM (2008) 426–434
11. Jin, R., Liu, L., Ding, B., Wang, H.: Distance-constraint reachability computation in uncertain graphs. Proc. VLDB Endow. **4** (2011) 551–562
12. Lin, C.J., Weng, R.C., Keerthi, S.S.: Trust region newton method for logistic regression. Journal of Machine Learning Research **9** (2008) 627–650
13. Witsenburg, T., Blockeel, H.: Improving the accuracy of similarity measures by using link information. In Kryszkiewicz, M., Rybinski, H., Skowron, A., Ras, Z.W., eds.: ISMIS. Volume 6804 of Lecture Notes in Computer Science., Springer (2011) 501–512
14. Taranto, C., Di Mauro, N., Esposito, F.: Probabilistic inference over image networks. Italian Research Conference on Digital Libraries 2011 CCIS **249** (2011) 1–13
15. Baccianella, S., Esuli, A., Sebastiani, F.: Evaluation measures for ordinal regression. In: Proceedings of the 2009 Ninth International Conference on Intelligent Systems Design and Applications. ISDA '09, IEEE Computer Society (2009) 283–287