Learning to Predict Web Collaborations

Lilyana Mihalkova
Dept. of Computer Science
University of Maryland
College Park, MD 20742 USA
lily@cs.umd.edu

Walaa Eldin Moustafa
Dept. of Computer Science
University of Maryland
College Park, MD 20742 USA
walaa@cs.umd.edu

Lise Getoor
Dept. of Computer Science
University of Maryland
College Park, MD 20742 USA
getoor@cs.umd.edu

ABSTRACT

Much of the knowledge available on the web today comes as a result of fruitful collaborations among large groups of people. One of the most striking examples of successful web collaboration is the online encyclopedia Wikipedia. The web is used as a collaboration platform by highly specialized blogging communities and by the scientific community. An important reason for the richness of content generated through web collaborations is that the participants in such collaborations are not constrained by geographic location. Thus, like-minded individuals from across the world can join their efforts. This also means, however, that web collaborators often do not know each other, and, thus, finding collaborators on the web is more difficult than it is with more traditional forms of collaboration that are initiated based on acquaintance. This difficulty is further exacerbated by the fact that web collaborations tend to be more dynamic as participants join and abandon communities. We consider the task of recommending project-specific potential collaborators to web users and propose an approach that is based on statistical relational learning. Our proposed model thus has the advantages that it can include complex features composed of multiple properties and relationships of the entities, it can handle the high levels of noise and uncertainty inherent in user actions, and it allows for joint decision-making, which leads to more accurate predictions. To ensure scalability, our model is trained in an online fashion. We demonstrate the effectiveness of our approach on a data set collected from Wikipedia.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

1. INTRODUCTION

A wealth of knowledge on a wide range of subjects is freely available on the web. Much of this content is being generated through the collaboration of large groups of people, and to a great extent, its richness is due to the fact that collaborations on the web are not constrained by geographic location. Thus, even users passionate about highly specific or unusual topics can form fruitful interactions with others who share their interests. One of the most striking examples of successful web collaboration is the Wikipedia online encyclopedia, where high quality comprehensive content is being generated and refined by "ordinary" people. Collaboration on the web happens also in more informal settings, such as in blogs. For example, a sizable community of crafts bloggers in geographically diverse locations are contributing a rich repository of techniques, patterns, and projects. While previously such knowledge was documented in books by single individuals or as "folklore" through personal acquaintances, now bloggers who have never met can build on each other's ideas, thus leading to more dynamic progress. The web is used as a collaboration platform also by the scientific community. For example, the peer review process is supported by web applications that allow researchers from all over the world to coordinate their reviewing efforts.

A characteristic typical of web collaborations that is due to their ability to transcend geographic constraints is that their participants frequently do not know each other. Unlike more traditional forms of collaboration where the members of a team know each other personally and often work in the same building, web collaborators frequently know each other only through aliases. Moreover, collaborations on the web tend to be more dynamic. Wikipedia editors come and go, new blogs get started, whereas others become defunct, and new members join the research community. Thus, the set of collaborators in a web community is in a constant state of flux. From the point of view of particular users, this means that they need to remain on the look-out for new potential collaborators to replace the ones that become inactive. These considerations motivate the need for modeling web users from the perspective of their collaborations so that new collaborators can be suggested to them automatically.

Problem Definition: Because web users often work on several possibly unrelated projects in parallel, it is essential to be able to match the ongoing projects of a user with her potential collaborators according to their interests. Thus, in this paper we study the task of recommending project-specific potential collaborators to a web user. We formalize this task as follows. There are two sets of entities: the set of users $\mathcal{U}$ and the set of projects $\mathcal{P}$. The set of projects of a user U is denoted by $\mathcal{P}_U \subset \mathcal{P}$. For a given user U and each of her projects $P \in \mathcal{P}_U$, the goal is to recommend as potential collaborators users from $\mathcal{U} \setminus \{U\}$ whose interests align closely with P. To make such recommendations, systems can use two sources of information – the intrinsic properties of each entity, as well as the relationships that connect entities. Thus, the sets $\mathcal{P}$ and $\mathcal{U}$ can be seen as annotated nodes in a large graph G, whose labeled edges represent relationships of different kinds. For example, two

projects may be connected by an edge to indicate that one of the projects references the other; an edge between two users may indicate that they exchanged messages; and a user can be linked to a project to indicate that the user is an active contributor to the project.

Recommendations for future collaborations can be significantly improved by taking advantage of these relationships in order to allow for predictions to be made collectively. For example, to assign potential collaborators to a given user U's projects, a system might consider each possible collaborator-project (C, P) pair independently of the others and determine its viability based on the characteristics shared between P and C's own projects $\mathcal{P}_C$, such as similar text or shared categories. However, by considering all (C, P) pairs jointly, accuracy can be improved. For instance, consider a situation where two of U's potential collaborators, C_1 and C_2, interact frequently. Then, because users who communicate often typically share interests, if we inferred that C_1 is relevant to a project P, we might infer that C_2 is also relevant to P, even if there was insufficient evidence to conclude this based just on the characteristics of P and C_2's projects. Similarly, based on spurious evidence, we may conclude that a potential collaborator C_3 is also relevant to P. However, if C_3 never communicates with any of the other users considered relevant to P, our belief in her relevance to P diminishes.

While the relational information in G can help make recommendations of significantly higher quality, it can also lead to approaches that do not scale well. In particular, techniques that perform reasoning over the entire G and require storing it all at the same time in memory are not applicable to large-scale web collaboration networks. To address this challenge, while keeping the benefits of relational information, our approach is trained and applied by considering G in pieces that are streamed one by one, where each piece is centered at one particular user in $\mathcal{U}$ and grown to a small depth around it.

We explore the task of recommending project-specific potential collaborators to a web user in the context of Wikipedia. In this setting $\mathcal{U}$ consists of the set of Wikipedia editors, and $\mathcal{P}$ corresponds to the set of articles in the encyclopedia. $\mathcal{P}_U$ is the set of articles to which a user U is contributing. Two users U_1 and U_2 collaborate if they both contribute to the same article(s), i.e., if $\mathcal{P}_{U_1} \cap \mathcal{P}_{U_2} \neq \emptyset$. The edges in G correspond to a variety of relationships. There are two kinds of edges between a user U and each of her articles $P \in \mathcal{P}_U$, depending on whether she directly edits P or participates in discussions regarding P's content; two users U_1 and U_2 are connected by an edge if they exchange messages, whereas a directed edge between two articles P_1 and P_2 indicates that they hyperlink to each other.

A variety of techniques for learning and reasoning in noisy and uncertain relational domains have been developed in the field of statistical relational learning (SRL) [5]. Highly expressive, SRL models can represent complex features composed of multiple properties and relationships of the entities. At the same time, because SRL models support probabilistic reasoning, they are well-suited to problems, such as those arising on the web, that involve using noisy evidence to make inherently uncertain predictions about user behavior. Our approach is based on one particular SRL model, Markov logic networks (MLNs) [24]. In MLNs, features are specified as first-order logic formulae, each of which has an attached weight that determines the formula's relative importance in the model. Formulae can be viewed as templates that are instantiated with the entities in a domain to define a Markov network over the prediction variables. We used MLNs because the first-order logic syntax of their formulae makes them intuitive to define and interpret and because a fairly mature implementation of MLNs is available [9].

For the task of project-specific collaboration recommendation, we defined a set of relational features as the formulae of an MLN. Each feature can be seen as describing a salient aspect of a situation in which two users begin collaborating on a particular Wikipedia article. To ensure the scalability of our method, the features are such that, to suggest collaborators for a particular user U, they need to be evaluated only in a small neighborhood of the relational graph G around U. Weights on the features are trained in an online fashion from instances that are streamed one at a time.

The remainder of this paper is structured as follows. Next we describe the data set used in the experiments and the representation we used to ensure scalability. As our approach is based on MLNs, in Section 3.1, we give MLN background and in Section 3.2, describe the relational features we define. In Section 4, we validate our model experimentally and demonstrate that the addition of global features that enforce collective decisions leads to significant gains in accuracy. We conclude with a discussion of related work.

2. DATA DESCRIPTION

We collected all Wikipedia articles that appeared in the featured[1] and controversial[2] lists in the period Oct. 7-21, 2009. These articles are interesting because they are richly connected, both by their hyperlinks and by their human network of editors [1]. In this way, we obtained a set $\mathcal{P}$ of 3,538 articles. For each article in $\mathcal{P}$, we collected the editors who contributed to it, either by directly editing the article, or by editing its "Talk," i.e., discussion, page. Only edits that were not marked as "minor" by the editor were considered. In this way, we obtained a set $\mathcal{U}$ of 280,068 editors. In addition, we collected the hyperlinks among the articles in $\mathcal{P}$. These articles are densely inter-linked, as indicated by the large number of hyperlinks (45,006) among them. Wikipedia articles often refer to external resources on the Web. In order to utilize the information in these external pages, for each article in $\mathcal{P}$, we looked up the categorizations of each of its external references in the DMOZ open directory[3]. Because this information is not available for all URLs, we considered both exact matches of URLs, for which there were about 0.9 per article in $\mathcal{P}$, as well as exact matches for just the domain name part of the URL, for which there were about 77 per article. An editor E_1 on Wikipedia can communicate with another editor E_2 by editing E_2's "Talk" page. Thus, we were able to collect data on the interactions among editors in $\mathcal{U}$. There were a total of 7,874,985 instances of communication between pairs of editors in $\mathcal{U}$.

In this work, we are interested in developing a scalable approach to learning from such richly relational data. To this end, we represented the data described above as a set of

[1] http://en.wikipedia.org/wiki/Wikipedia:Featured_lists

[2] http://en.wikipedia.org/wiki/Wikipedia:List_of_controversial_issues

[3] http://www.dmoz.org/

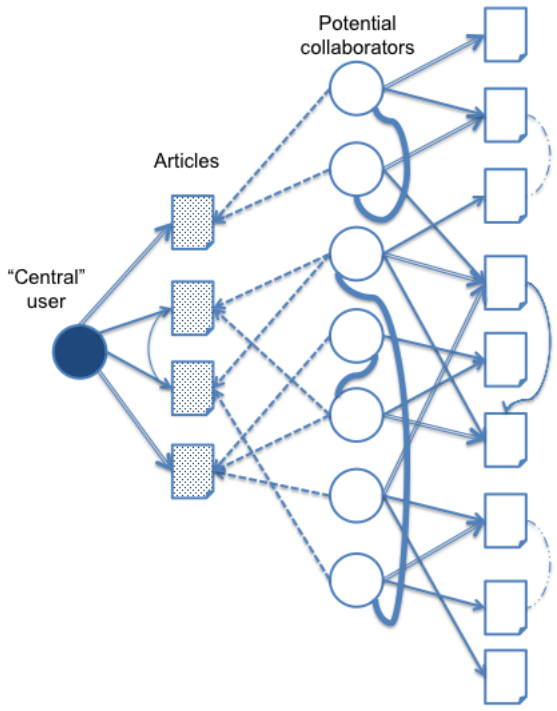

Figure 1: Sketch of a user-centered subgraph

relational subgraphs, where each subgraph G_C is centered around one of the editors C from $\mathcal{U}$. Each train/test instance consists of a single subgraph. Figure 1 shows a sketch of such a subgraph. The dark circle on the left represents the "central" user C from whose perspective the subgraph is constructed. G_C contains all articles $\mathcal{P}_C$ on which C is currently working, shown in the figure as the the dotted rectangles to which the dark circle points. There are two types of directed edges from a user to an article, depending on whether the user directly edited an article or edited the discussion page associated with it. Each subgraph additionally includes the set of editors $\mathcal{U}_C$, shown as white circles, who are contributing to at least one of the articles in $\mathcal{P}_C$, and for each of these editors, the set of other articles to which they are contributing, shown as white rectangles to which the white circles point. G_C also contains undirected edges between pairs of editors in $\mathcal{U}_C \cup \{C\}$ who have communicated with one another, and a variety of edges between pairs of articles to indicate relationships such as hyperlinking to one another, sharing categories, or having similar text.

Table 1 gives a complete list of the predicates used to specify G_C. Observed edits of articles or articles' discussion pages are captured by the `topicEdit` and `topicTalk` predicates respectively for non-central users, and the `centralEdit` and `centralTalk` predicates for the central user of a subgraph. The `similar` and `verySimilar` predicates are pre-computed based on the existing text of the two articles at the time when we collected the data. The tf-idf weighted representation of each article was constructed using the standard procedure of first performing stemming on the text (using the Porter stemmer), then removing stop words [15]. The `cat` predicate gives the category of each article, where the categories correspond to the section headings under which articles were listed on the controversial and featured article lists. There were a total of 19 categories.

To evaluate our model, at test time the links between $\mathcal{U}_C$ and $\mathcal{P}_C$, corresponding to dashed edges in Figure 1, are hidden, and the task is to predict them. Because we are interested in whether or not a collaboration forms, and not in the exact format of the collaboration, for the hidden links we do not distinguish between directly editing an article and editing its discussion page. Thus, all hidden links are represented by the `modifies` relation, regardless of whether a user contributed directly to an article or to its discussion page. At train time, fully observed subgraphs are provided as data.

Testing can be set up in a variety of ways. For example, one could hide only some of the edges between $\mathcal{U}_C$ and $\mathcal{P}_C$ and provide the rest as evidence. This corresponds to a scenario where the central user has some existing collaborations, and we would like to recommend new ones. However, because of the dynamic nature of web collaborations, where the set of active users is in a constant state of flux and new projects can be initiated by individual users, we would like to ensure that our approach does not depend on observed previous interactions of the potential collaborators with the projects of the central entity. These considerations have motivated the adoption of the more challenging test set-up described above, where all of the edges between $\mathcal{U}_C$ and $\mathcal{P}_C$ are hidden.

3. LEARNING TO PREDICT COLLABORATIONS

Our proposed model for project-specific collaborator recommendation is represented as a Markov logic network (MLN) [24]. Before describing the model, we provide background on MLNs.

3.1 Background

A Markov logic network (MLN) [24] consists of a set of first-order logic formulae $\mathcal{F}$, each of which has an associated weight. MLNs can be viewed as relational analogs to Markov networks, in which the potential functions over cliques are defined by the groundings of the formulae in $\mathcal{F}$. The role of first-order logic, therefore, is to provide a highly expressive language for specifying general relational features. This is particularly appealing in our task because the domain contains several different relationships (as defined in Table 1) among entities of different types (e.g., users and projects), and first-order logic is a natural representation for features composed of such relationships.

MLNs are appealing as a representation for our task also because, once their first-order logic formulae are grounded with a given set of entities (i.e., particular set of users and projects), they define a Markov network, which provides principled support for probabilistic reasoning. In particular, an MLN computes the conditional joint probability of a set of unknown predicate groundings $\mathbf{X}$, given truth values for a set of evidence predicate groundings $\mathbf{Y}$ as follows:

$$P(\mathbf{X} = \mathbf{x} | \mathbf{Y} = \mathbf{y}) = \frac{\exp(\sum_{f_i \in \mathcal{F}} w_i n_i(\mathbf{x}, \mathbf{y}))}{\sum_{\mathbf{x}'} \exp(\sum_{f_i \in \mathcal{F}} w_i n_i(\mathbf{x}', \mathbf{y}))} \quad (1)$$

Above, $\mathbf{X}$ and $\mathbf{Y}$ are the sets of all unknown and evidence groundings, respectively, of the predicates in the domain; $\mathbf{x}$ and $\mathbf{y}$ are the sets of corresponding truth assignments; w_i is the weight associated with formula f_i; $n_i(\mathbf{x}, \mathbf{y})$ is the

Table 1: Predicates used to specify a user-centered subgraph.

`modifies(A, U)`	Represents the relationship predicted at test time.
`topicEdit(A, U)`	True iff user U directly edited article A.
`topicTalk(A, U)`	True iff user U contributed to the discussion about article A.
`centralEdit(A)`	True iff the central user of the instance edited article A.
`centralTalk(A)`	True iff the central user of the instance contributed to the discussion about article A.
`userTalk(U1, U2)`	True iff $U1$ and $U2$ talked to each other.
`hyperLink(A1, A2)`	True iff there is a hyperlink from article $A1$ to article $A2$.
`similar(A1, A2)`	True iff the cosine similarity of the tf-idf weighted bag-of-word vector representations of $A1$ and $A2$ is > 0.1 but ≤ 0.5.
`verySimilar(A1, A2)`	True iff the cosine similarity of the tf-idf weighted bag-of-word vector representations of $A1$ and $A2$ is > 0.5.
`cat(A, C)`	True iff article A has category C. The category of each article is the heading under which it is listed on the controversial or featured articles lists.
`level1Exact(A, EC)`	True iff an external link listed on A has category EC in the first level of the DMOZ hierarchy.
`level2Exact(A, EC)`	True iff an external link listed on A has category EC in the second level of the DMOZ hierarchy.
`level1Inexact(A, EC)`	True iff the domain name part of an external link listed on A has category EC in the first level of the DMOZ hierarchy.
`level2Inexact(A, EC)`	True iff the domain name part of an external link listed on A has category EC in the second level of the DMOZ hierarchy.

number of `true` groundings of formula f_i on truth assignment $\mathbf{x}, \mathbf{y}$; and the denominator computes the normalizing partition function Z.

Because evaluating the j-th grounding g_i^j of formula f_i requires us to consider only the truth values in $\mathbf{x}, \mathbf{y}$ that correspond to the grounded literals[4] that appear in g_i^j, each grounding g_i^j defines a clique over the grounded literals that appear in it. The value of the potential function over this clique is $\exp(w_i \mathbb{1}(g_i^j(\mathbf{x}, \mathbf{y}) == \texttt{true}))$, where $\mathbb{1}(A)$ returns 1 if A evaluates to `true` and 0 otherwise.

To illustrate, consider a simple MLN that contains a single formula with weight W:

$$W : \texttt{topicEdit}(\texttt{A}, \texttt{U}) \wedge \texttt{similar}(\texttt{A}, \texttt{B}) \Rightarrow \texttt{modifies}(\texttt{B}, \texttt{U}) \quad (2)$$

This formula states that if a user U previously edited an article A that is similar to one of the existing projects B of the central user, then U will collaborate on B.

To draw inferences about a particular set of entities, all groundings of this formula with the given entities are generated. For example, consider a toy scenario in which the central user is working on two projects, `hong_kong` and `soccer`, there is a single potential collaborator, `john`, who edits a single existing project, `china`. For the sake of illustration, imagine that
$\texttt{similar}(\texttt{china}, \texttt{hong\_kong}) == \texttt{true}$ and
$\texttt{similar}(\texttt{china}, \texttt{soccer}) == \texttt{false}$. The relational subgraph corresponding to this data set is shown in Figure 2 a). The central user is implicit in the subgraph – while it appears in the figure, we do not need to name it explicitly. Given these entities, Formula 2 has the following groundings:

$$\texttt{topicEdit}(\texttt{china}, \texttt{john}) \wedge \texttt{similar}(\texttt{china}, \texttt{hong\_kong}) \Rightarrow \texttt{modifies}(\texttt{hong\_kong}, \texttt{john}) \quad (3)$$

$$\texttt{topicEdit}(\texttt{china}, \texttt{john}) \wedge \texttt{similar}(\texttt{china}, \texttt{soccer}) \Rightarrow \texttt{modifies}(\texttt{soccer}, \texttt{john}) \quad (4)$$

Note that we did not form any groundings with $\texttt{topicEdit}(\texttt{soccer}, \texttt{john})$ or $\texttt{topicEdit}(\texttt{hong\_kong}, \texttt{john})$. This is because, for convenience in this work we consider the projects of the central user and those of the potential collaborators to be of different types. This is indicated in Figure 1 by the different shading used for the articles edited just by the potential collaborators and those edited also by the central user.

The set $\mathbf{X}$ contains the grounded literals $\texttt{modifies}(\texttt{hong\_kong}, \texttt{john})$ and $\texttt{modifies}(\texttt{soccer}, \texttt{john})$, and the set $\mathbf{Y}$ contains the remaining grounded literals in formulae 3 and 4 above. Because the truth values of grounded literals in $\mathbf{Y}$ are known, we can simplify the above grounded formulae. Formula 3 becomes
$\texttt{true} \wedge \texttt{true} \Rightarrow \texttt{modifies}(\texttt{hong\_kong}, \texttt{john})$, which can be further simplified to $\texttt{modifies}(\texttt{hong\_kong}, \texttt{john})$. Formula 4 becomes $\texttt{true} \wedge \texttt{false} \Rightarrow \texttt{modifies}(\texttt{soccer}, \texttt{john})$. This formula evaluates to `true` regardless of the value assigned to the unknown ground literal $\texttt{modifies}(\texttt{soccer}, \texttt{john})$. Because such formulae cancel from the numerator and denominator in Equation 1, they can be safely ignored.

The resulting Markov network contains two disconnected nodes, corresponding to $\texttt{modifies}(\texttt{hong\_kong}, \texttt{john})$ and $\texttt{modifies}(\texttt{soccer}, \texttt{john})$ respectively, as shown in Figure 2 b). The potential function for $\texttt{modifies}(\texttt{hong\_kong}, \texttt{john})$ is computed as $\exp(W\mathbb{1}(\texttt{modifies}(\texttt{hong\_kong}, \texttt{john}) == \texttt{true}))$, whereas the potential function for $\texttt{modifies}(\texttt{soccer}, \texttt{john})$ is computed as $\exp(0)$. Thus, in our simple toy example, the Markov network is a logistic regression model whose features consist of the groundings of Formula 2. In fact, an MLN defines a logistic regression model whenever each

[4]A literal is a negated or non-negated atom, where an atom is defined as a predicate applied to variables or entities in the domain. A literal is grounded if it contains only entities. For example, `modifies(hong_kong, john)` is a grounded literal, whereas `modifies(A, U)` is ungrounded.

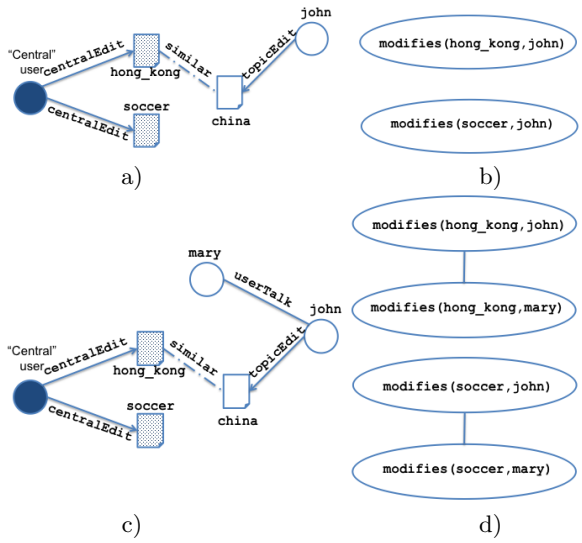

Figure 2: Toy example subgraphs (a) and (c), and their corresponding grounded Markov networks (b) and (d)

of its formulae contains at most a single unknown literal (see also http://alchemy.cs.washington.edu/tutorial/4Logistic_Regression.html). The advantage of using MLNs in this case comes from the convenience of defining general relational features that are not tied to specific entities.

Further advantages are obtained if we use formulae that contain more than one unknown literal. Consider, for example, the formula:

$$W' : \texttt{userTalk}(\texttt{U1},\texttt{U2}) \wedge \texttt{modifies}(\texttt{A},\texttt{U1}) \wedge \texttt{modifies}(\texttt{A},\texttt{U2}) \quad (5)$$

This formula suggests that if two users talked to each other, then they would be relevant collaborators on the same projects. Continuing the toy example, we introduce a second potential collaborator, `mary`, where $\texttt{userTalk}(\texttt{john},\texttt{mary}) = \texttt{true}$, and all other groundings of the `userTalk` predicate are `false`. This is shown in Figure 2 c). Analogous to the example above, groundings of Formula 5 in which the `userTalk` literal is `false` evaluate to `false` regardless of the values of the `modifies` literals and can therefore be dropped from the model because they cancel from the numerator and denominator of Equation 1. Thus, the result of adding this formula to our MLN is that now the Markov network obtained after grounding will contain the cliques $\{\texttt{modifies}(\texttt{hong\_kong},\texttt{john}), \texttt{modifies}(\texttt{hong\_kong},\texttt{mary})\}$ and $\{\texttt{modifies}(\texttt{soccer},\texttt{john}), \texttt{modifies}(\texttt{soccer},\texttt{mary})\}$, as illustrated in Figure 2 d). Thus, unlike in the logistic regression model we obtained earlier, now our model is able to reason about joint assignments of truth values to the unknown ground literals.

A fairly mature implementation of MLNs is provided in the Alchemy system [9], in which several algorithms are available for performing learning and inference. For parameter learning, we used the contrastive divergence algorithm, described in [14], which can be seen as a voted-perceptron-like gradient descent algorithm, in which the gradient for updating the weight of a formula F is computed as the difference between the number of true groundings of F in the data and the expected number of true groundings of F according to the currently learned weights, where the expectation is computed by carrying out a small number of MCMC steps over the model. We chose this algorithm because computationally it is relatively cheap, and adapted the existing implementation in Alchemy such that the data subgraphs can be streamed one at a time to the learner. For inference, we used the MC-SAT algorithm, which has been shown to be very accurate and efficient [20].

3.2 MLNs for Collaborator Recommendation

We address the problem of project-specific collaborator recommendation by hand-engineering a set of relational features as the formulae in an MLN and training weights for these formulae on available data. We introduce two models. All rules in the first model contain only a single literal of the unknown predicate `modifies`. Thus, as discussed in Section 3.1, this model corresponds to a logistic regression classifier whose (Boolean) features are the groundings of the formulae. We call this model **Local** because its decisions are local to each unknown. The second model additionally includes a small set of formulae that contain more than one literal of the unknown predicate and thus allows for reasoning over joint assignments of truth values to the unknown grounded literals. This model is called **Global**, and, as shown in the experiments, its predictions are significantly more accurate than those of **Local.**

Table 2 shows all formulae in the **Local** model. The predicates used in the formulae are as described in Table 1, except that some of the predicates have different versions that indicate the type of article they take as an argument. For example, the `bqSimilar` predicate takes as a first argument a "background" article, i.e., one of the white ones in Figure 1, and as a second argument a "query" article, i.e., one of the dotted ones in Figure 1. The first 8 formulae in Table 2 state that a user U will collaborate with the central user on articles that are in some way related to articles edited by U. Each of these first 8 formulae is concerned with a different way in which two articles can be related, by virtue of sharing content, hyperlinking to one another, or sharing categories. The second set of 8 formulas is analogous, except that it relates the articles of the central user to articles to which U contributed by editing the discussion page.

Table 3 shows the additional set of formulae included in the **Global** model in order to allow for joint decisions. These formulae are analogous to Formula 5 used for illustration in Section 3.1 and establish different kinds of correlations between the relevance of two potential collaborators, who interact among each other, to the same article. The first two formulae assert that two potential collaborators who talk to each other should be relevant or irrelevant to the same projects of the central user. The third formula states that one of the collaborators should be relevant, while the other one should be irrelevant. This last formula seems to contradict our intuitions; however, it is important to remember that weight learning can assign a negative weight to it, in which case, during inference, truth assignments that contradict it would have higher probability than truth assignments that agree with it.

4. EXPERIMENTS

We compared the performance of the **Local** model, which can be viewed as a logistic regression baseline, to that of the **Global** model on the data set described in Section 2.

Table 2: Formulae in the Local model

$\texttt{topicEdit}(\texttt{T},\texttt{U}) \wedge \texttt{bqSimilar}(\texttt{T},\texttt{T1}) \Rightarrow \texttt{modifies}(\texttt{T1},\texttt{U})$
$\texttt{topicEdit}(\texttt{T},\texttt{U}) \wedge \texttt{bqVerySimilar}(\texttt{T},\texttt{T1}) \Rightarrow \texttt{modifies}(\texttt{T1},\texttt{U})$
$\texttt{topicEdit}(\texttt{T},\texttt{U}) \wedge \texttt{bqHyperLink}(\texttt{T},\texttt{T1}) \Rightarrow \texttt{modifies}(\texttt{T1},\texttt{I})$
$\texttt{topicEdit}(\texttt{T},\texttt{U}) \wedge \texttt{bCat}(\texttt{T},\texttt{C}) \wedge \texttt{qCat}(\texttt{T1},\texttt{C}) \Rightarrow \texttt{modifies}(\texttt{T1},\texttt{U})$
$\texttt{topicEdit}(\texttt{T},\texttt{U}) \wedge \texttt{bLevel1Exact}(\texttt{T},\texttt{C}) \wedge \texttt{qLevel1Exact}(\texttt{T1},\texttt{C}) \Rightarrow \texttt{modifies}(\texttt{T1},\texttt{U})$
$\texttt{topicEdit}(\texttt{T},\texttt{U}) \wedge \texttt{bLevel2Exact}(\texttt{T},\texttt{C}) \wedge \texttt{qLevel2Exact}(\texttt{T1},\texttt{C}) \Rightarrow \texttt{modifies}(\texttt{T1},\texttt{U})$
$\texttt{topicEdit}(\texttt{T},\texttt{U}) \wedge \texttt{bLevel1Exact}(\texttt{T},\texttt{C}) \wedge \texttt{qLevel1Inexact}(\texttt{T1},\texttt{C}) \Rightarrow \texttt{modifies}(\texttt{T1},\texttt{U})$
$\texttt{topicEdit}(\texttt{T},\texttt{U}) \wedge \texttt{bLevel2Exact}(\texttt{T},\texttt{C}) \wedge \texttt{qLevel2Inexact}(\texttt{T1},\texttt{C}) \Rightarrow \texttt{modifies}(\texttt{T1},\texttt{U})$
$\texttt{topicTalk}(\texttt{T},\texttt{U}) \wedge \texttt{bqSimilar}(\texttt{T},\texttt{T1}) \Rightarrow \texttt{modifies}(\texttt{T1},\texttt{U})$
$\texttt{topicTalk}(\texttt{T},\texttt{U}) \wedge \texttt{bqVerySimilar}(\texttt{T},\texttt{T1}) \Rightarrow \texttt{modifies}(\texttt{T1},\texttt{U})$
$\texttt{topicTalk}(\texttt{T},\texttt{U}) \wedge \texttt{bqHyperLink}(\texttt{T},\texttt{T1}) \Rightarrow \texttt{modifies}(\texttt{T1},\texttt{U})$
$\texttt{topicTalk}(\texttt{T},\texttt{U}) \wedge \texttt{bCat}(\texttt{T},\texttt{C}) \wedge \texttt{qCat}(\texttt{T1},\texttt{C}) \Rightarrow \texttt{modifies}(\texttt{T1},\texttt{U})$
$\texttt{topicTalk}(\texttt{T},\texttt{U}) \wedge \texttt{bLevel1Exact}(\texttt{T},\texttt{C}) \wedge \texttt{qLevel1Exact}(\texttt{T1},\texttt{C}) \Rightarrow \texttt{modifies}(\texttt{T1},\texttt{U})$
$\texttt{topicTalk}(\texttt{T},\texttt{U}) \wedge \texttt{bLevel2Exact}(\texttt{T},\texttt{C}) \wedge \texttt{qLevel2Exact}(\texttt{T1},\texttt{C}) \Rightarrow \texttt{modifies}(\texttt{T1},\texttt{U})$
$\texttt{topicTalk}(\texttt{T},\texttt{U}) \wedge \texttt{bLevel1Exact}(\texttt{T},\texttt{C}) \wedge \texttt{qLevel1Inexact}(\texttt{T1},\texttt{C}) \Rightarrow \texttt{modifies}(\texttt{T1},\texttt{U})$
$\texttt{topicTalk}(\texttt{T},\texttt{U}) \wedge \texttt{bLevel2Exact}(\texttt{T},\texttt{C}) \wedge \texttt{qLevel2Inexact}(\texttt{T1},\texttt{C}) \Rightarrow \texttt{modifies}(\texttt{T1},\texttt{U})$

Table 3: Formulae in the Global model

All formulae from Table 2.
$\texttt{userTalk}(\texttt{U},\texttt{V}) \wedge \texttt{modifies}(\texttt{T},\texttt{U}) \wedge \texttt{modifies}(\texttt{T},\texttt{V})$
$\texttt{userTalk}(\texttt{U},\texttt{V}) \wedge !\texttt{modifies}(\texttt{T},\texttt{U}) \wedge !\texttt{modifies}(\texttt{T},\texttt{V})$
$\texttt{userTalk}(\texttt{U},\texttt{V}) \wedge \texttt{modifies}(\texttt{T},\texttt{U}) \wedge !\texttt{modifies}(\texttt{T},\texttt{V})$

Additionally, we present the performance of a baseline that predicts randomly.

4.1 Methodology

Training and testing was performed on subgraphs centered around users who made edits to the encyclopedia on at least 30 distinct days, had at least 30 collaborators, and edited at most 15 different articles. These restrictions are motivated by the observation that collaborator suggestion is most needed by editors who are strongly engaged with the encyclopedia, and so contribute to it over extended periods, but at the same time are focused in their interests. In this way, we exclude users, such as the "60% of registered users [who] never make another edit after their first 24 hours" [19], as well as users who help oversee the editing process and are therefore somewhat superficially involved in large numbers of edits, from being central users. However, such users can still appear as potential collaborators, i.e., as the white circles in Figure 1. We obtained a total of 1785 subgraphs.

Four-fold cross-validation was performed by splitting the 1785 subgraphs in our data randomly into 4 folds and performing 4 train/test runs, in each run withholding one of the folds for testing, and training on the remaining three. Fully observed data is provided during training. As discussed in Section 2, during testing we hide the truth values of all groundings of the `modifies` predicate and test the models on how well they predict them.

To evaluate a model, we rank the potential project-collaborator $(\texttt{p},\texttt{c})$ pairs according to the probability predicted for the $\texttt{modifies}(\texttt{p},\texttt{c})$ literals by the model, and use two standard metrics from the information retrieval literature [15]:

- (MAP) Mean average precision, which is identical to the area under the precision-recall curve. The MAP score is computed over a set of test subgraphs $\mathcal{S}$ as follows:

$$\text{MAP}(\mathcal{S}) = \frac{1}{|\mathcal{S}|}\sum_{s\in\mathcal{S}}\frac{1}{|R_s|}\sum_{r\in R_s} P@r.$$

Above, R_s is the set of all possible $(\texttt{p},\texttt{c})$ pairs, and the precision at r is defined as

$$P@r = \frac{\text{Num of true positive pairs among the top } r}{r}$$

- (AUC-ROC) Area under the ROC Curve, which is identical to the mean average true negative rate. This score is computed as follows:

$$\text{AUC-ROC}(\mathcal{S}) = \frac{1}{|\mathcal{S}|}\sum_{s\in\mathcal{S}}\frac{1}{|R_s|}\sum_{r\in R_s} TN@r,$$

where the true negative rate at r is defined as

$$TN@r = \frac{\text{Number of true negatives below position } r}{\text{Total num true negatives}}.$$

4.2 Results

Figure 3 shows the experimental results. The observed differences are significant at the 0.001 level according to a paired t-test, except for the difference in AUC-ROC between **Random** and **Local**. **Local** provides a small advantage over **Random** in terms of MAP score, but its performance is statistically indistinguishable from **Random** in terms of AUC-ROC. On the other hand, the addition of the three simple formulae from Table 3 leads to significant performance gains on both scores, demonstrating the usefulness of **Global**'s collective predictions in this domain.

Training of both the **Local** and **Global** models was very efficient. Using dedicated Intel Xeon 2.67GHz CPUs, average training time per subgraph on average was 0.11 seconds for **Local** and 0.21 seconds for **Global**.

5. RELATED WORK

The problem of collaborator suggestion has been addressed by several other authors, although, to the best of our knowledge, previous work does not specifically target the setting of highly dynamic collaborations on the Web. Existing work on this topic has focused on an academic setting. Zaiane

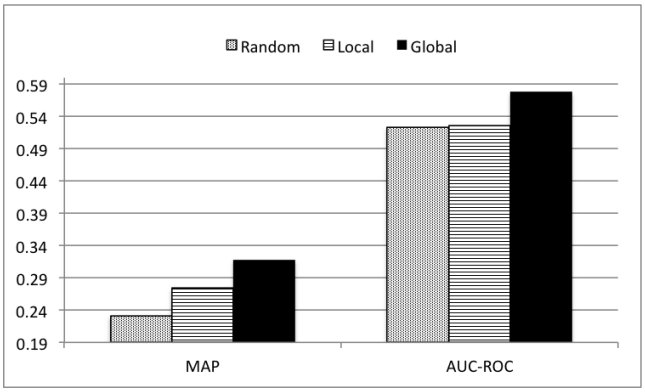

Figure 3: Accuracy of the different models. All differences are significant at the 0.001 level, except for the difference in AUC-ROC between Local and Random.

et al. [25] use DBLP data to recommend research collaborations and find research communities by developing a random walk algorithm that is based on a tripartite representation of DBLP data consisting of author, conference, and topic information. Relevancy scores are computed by running an extension of the transitional random walk algorithm over this tripartite graph. In [13], relevancy scores are derived based on two metrics – correlation and cooperation. Correlation is expressed in terms of two authors' mutual interest in different research areas, and cooperation is expressed in terms of their common collaborators. In [11], the task of predicting author collaborations is cast as link prediction in social networks. A variety of social network analysis measures, such as common neighbors, Jaccard coefficient, PageRank, and Hitting time, are proposed and evaluated according to their accuracy in predicting collaborations.

There are two main difficulties associated with applying these techniques in our setting. First, our focus is on approaches that, unlike techniques based on random walks, do not need to perform computations over the entire collaboration graph, but, rather, compute features only over small subgraphs centered around the user for whom recommendations are made. Such approaches are more likely to scale in the much more dynamic Web collaboration setting. Second, in order to ensure that our model can address the most challenging, worst-case, scenario in which a central user's previous collaborators have all become inactive (not an unlikely event in a web setting), during testing, we hide all links between a central user's articles and her potential collaborators, thus simulating the case where a new set of collaborators is suggested. For this reason, techniques that require observations of previous shared projects between users cannot be applied. Furthermore, unlike the approach presented here, techniques that make recommendations based on relevance scores do not involve learning and use only structural features, not taking into account attributes of the authors or the document contents. In contrast, our approach can incorporate a variety of structural and content features and learn the relative importance of such features. Finally, the approaches outlined above do not make joint recommendations, which, as shown in Section 4, can lead to significant gains in accuracy.

The approach of Gunawardena and Weber [7] finds prototypical collaborations by mining previous successful collaboration experiences. Prototypical collaborations are abstractions of collaborations at the level of subject area, e.g., Mathematician + Computer Scientist. After inferring a database of such prototypical collaborations, actual collaborators are found by utilizing expert locating systems. This approach is much better-suited to scientific collaboration than to other domains such as Wikipedia, as authors do not usually list their profession, in addition to the fact that Wikipedia is based on the idea that it can be edited by ordinary people and is not exclusive to professionals.

A related problem is that of team formation. For example, Lappas et al. [10] study this problem in the setting where social network information is available and show how to use this additional information to assemble a team whose members can work together well. While related, research on team formation differs from the problem addressed here in several important aspects. First, the goal of team formation is to assemble a group of collaborators for a single project; in contrast, our focus is on suggesting collaborators for all the projects of a user simultaneously. Second, in our setting no explicit information on the skills of potential collaborators is available. Finally, in contrast to the work of Lappas et al. [10], where social network information is used to enhance team cohesion, here interactions among users indicate shared interests.

On the surface, the problem studied here is closely aligned to the task addressed by collaborative filtering. Collaborative filtering has been successfully applied to predict a user's rating for an item by observing a partial list of her ratings [6, 23, 2, 8, 12]. Unobserved ratings are predicted by employing ratings of like-minded users, where the set of like-minded users is determined by correlating the similarity of their ratings with the current user's ratings. One could try to cast project-specific collaborator recommendation as a collaborative filtering task in two ways. The first approach is to view the potential collaborators as "items." In this scenario, the goal could be to predict the rating given by the central user to each of these "items." This formulation, however, does not allow for project-specific recommendations. Moreover, it requires observations of previous shared collaborators between the central entity and other users, who also collaborated with some of the "items." Such observations are not available in our case. A second approach could be to treat the (dotted in Figure 1) articles of the central user as "items" and the potential collaborators as users whose rating for each of the items we would like to predict. While such an approach would allow for project-specific recommendations, applying it requires that at least some of the (dashed in Figure 1) links between the central user's articles and the potential collaborators be observed. To overcome the need for previously observed collaborations, our approach compares articles based on their content and other features. In this respect, our approach is more akin to techniques that also use content-based information to make recommendations, e.g., [22, 16]. The model proposed here, however, additionally incorporates various structural features, as well as interactions among the users, which allow for joint predictions.

Our work builds on the extensive literature on collective classification, relational data mining, and statistical relational learning, e.g., [3, 18, 5, 4]. The proposed approach is most closely related to previous applications of MLNs, e.g., [21, 17]. Our results confirm the conclusions drawn in

a variety of previous tasks in which joint predictions based on relational information were significantly more accurate. In this paper, we are able to maintain these benefits of collective prediction while trimming training subgraphs to a reasonable size.

6. CONCLUSIONS AND FUTURE WORK

This paper addresses the task of project-specific collaborator recommendation on the web. Our approach is implemented as an MLN and benefits from the ability to define complex relational features and to perform principled probabilistic inference over joint predictions. At the same time, training is performed in an online fashion, considering only small subgraphs of the domain at any given time step. We evaluated the proposed approach on a data set collected from the Wikipedia online encyclopedia and demonstrated that the ability to perform joint predictions is essential in this setting. Avenues for future work include evaluating analogous models in other web collaboration settings, e.g., among bloggers, and investigating the usefulness of additional relational features.

Acknowledgment

We thank the anonymous reviewers for their comments. LM is supported by a CI Fellowship under NSF Grant #0937060 to the Computing Research Association.

7. REFERENCES

[1] U. Brandes, P. Kenis, J. Lerner, and D. van Raaij. Network analysis of collaboration structure in Wikipedia. In *Proceedings of WWW*, 2009.

[2] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. Technical Report MSR-TR-98-12, Microsoft Research, 1998.

[3] S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *Proceedings of SIGMOD*, 1998.

[4] L. De Raedt. *Logical and Relational Learning*. Springer Verlag, Berlin, 2008.

[5] L. Getoor and B. Taskar, editors. *Introduction to Statistical Relational Learning*. MIT Press, Cambridge, MA, 2007.

[6] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35:61–70, December 1992.

[7] S. Gunawardena and R. Weber. Discovering patterns of collaboration for recommendation. In *Proceedings FLAIRS*, 2009.

[8] J. Herlocker, J. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of SIGIR*, 1999.

[9] S. Kok, P. Singla, M. Richardson, and P. Domingos. The Alchemy system for statistical relational AI. Technical report, Department of Computer Science and Engineering, University of Washington, 2005. `http://www.cs.washington.edu/ai/alchemy`.

[10] T. Lappas, K. Liu, and E. Terzi. Finding a team of experts in social networks. In *Proceedings of KDD*, 2009.

[11] D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. In *Proceedings of CIKM*, 2003.

[12] G. Linden, B. Smith, and J. York. Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.

[13] G. R. Lopes, M. M. Moro, L. K. Wives, and J. P. M. de Oliveira. Collaboration recommendation on academic social networks. In *Advances in Conceptual Modeling – Applications and Challenges ER Workshops*, 2010.

[14] D. Lowd and P. Domingos. Efficient weight learning for Markov logic networks. In *Proceedings of PKDD*, 2007.

[15] C. D. Manning, P. Raghavan, and H. Schutze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[16] P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Proceedings of AAAI*, 2002.

[17] L. Mihalkova and R. J. Mooney. Learning to disambiguate search queries from short sessions. In *Proceedings of ECML/PKDD*, 2009.

[18] J. Neville and D. Jensen. Iterative classification in relational data. In *Proceedings of the SRL Workshop at AAAI*, 2000.

[19] K. Panciera, A. Halfaker, and L. Terveen. Wikipedians are born, not made: a study of power editors on Wikipedia. In *Proceedings of GROUP*, 2009.

[20] H. Poon and P. Domingos. Sound and efficient inference with probabilistic and deterministic dependencies. In *Proceedings of AAAI*, 2006.

[21] H. Poon and P. Domingos. Joint inference in information extraction. In *Proceedings of AAAI*, 2007.

[22] A. Popescul, L. H. Ungar, D. M. Pennock, and S. Lawrence. Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In *Proceedings of UAI*, 2001.

[23] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. GroupLens: an open architecture for collaborative filtering of netnews. In *Proceedings of CSCW*, 1994.

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

[25] O. R. Zaiane, J. Chen, and R. Goebel. DBconnect: mining research community on DBLP data. In *Proceedings of WebKDD/SNA-KDD*, 2007.