A Framework for Co-Classification of Articles and Users in Wikipedia

Lei Liu
Dept. of Computer Science & Engineering
Michigan State University
East Lansing, US
Email: liulei1@msu.com

Pang-Ning Tan
Dept. of Computer Science & Engineering
Michigan State University
East Lansing, US
Email: ptan@msu.edu

***Abstract*—The massive size of Wikipedia and the ease with which its content can be created and edited has made automatic classification of Wikipedia articles and/or users an important research problem. Examples of Wikipedia classification tasks include categorizing users based on the articles they have edited and detecting content that should be flagged as spam, vandalism, or controversial/inflammatory articles. These tasks are typically cast into a network classification problem, in which the class label of an article or a user is determined based on their features (such as words that appear in an article or a user page) and links to other articles (or users) in the network. Previous works have focused mostly on a single-network, single-classification task. Yet there are many situations in which the network classification can be aided by knowing collectively the class labels of the users and articles (e.g., spammers are more likely to add spam content than non-spammers). Towards this end, we propose a novel framework that jointly classifies the Wikipedia articles and users, assuming there are correspondences between their classes. Our experimental results demonstrate that the proposed algorithm improves the classification performance for all the classes in Wikipedia article and user networks.**

***Keywords*-Networks classification; Wikipedia**

I. Introduction

Wikipedia[1] has become the largest online knowledge repository with approximately 15 million articles written in more than 270 languages, among which more than 3 million of them are in English. These articles are written collaboratively by voluntary users around the world. Given its massive amount of information and the fact that most of its content can be edited freely by any users who have access to the Internet, it is not surprising Wikipedia has become a rich domain for a variety of interesting classification problems such as topic detection of Wikipedia articles, categorization of Wikipedia editors as humans or bots, and classification of Wikipedia content as featured articles, controversial/inflammatory, spam, or vandalism. These tasks can be cast into a network classification problem, in which the goal is to assign a class label to each node in the network (e.g., an article or a user) based on the node features and link information [1][2][5][21].

[1]http://en.wikipedia.org

Most of the previous works have focused on classifying nodes in a single network. Since the Wikipedia articles and users are inextricably related, knowing the class label of an article often aids in the classification of users, and vice-versa. For example, in topic detection of Wikipedia articles, the category of an article provides evidence for the interest area of a user who edited the article's content. Similarly, in spam detection, we expect spam content to be more likely contributed by spammers than non-spammers. Assigning class labels to both articles and users in Wikipedia not only increases the amount of labeled data available, it is often a natural process during the creation of the training data itself (e.g., to label a user as a spammer or a vandal, one has to know whether the edited content should be considered a form of spam or vandalism). Despite the obvious relationship between their classes, none of the previous works on Wikipedia classification [14][11][7] were designed to co-classify the article and user networks.

We had previously developed a co-classification method to detect spam and spammers in social bookmarking Web sites [4]. Our method is based on extending the least-square support vector machine (LS-SVM) formulation to network data, taking into account the correspondences between the classes in different networks. Nevertheless, the method assumes a binary class problem, where the nodes in each network are assigned to one of two possible classes (spam/non-spam or spammer/non-spammer). For the more general Wikipedia classification problems, the nodes in the user and article networks may have more than two classes. Though it may be possible to transform the multi-class problem into multiple binary classification problems (using the 1-vs-1 or 1-vs-rest strategies), this approach may not be effective because each binary classifier is trained independently, disregarding the effectiveness of other classifiers. It is also inapplicable when the number of classes in article and user networks are different.

Our contributions in this paper are two-fold. First, we present an extension of the LS-SVM formulation for network data to a multi-class problem. Specifically, we formalize the joint classification tasks as a constrained optimization problem, in which the relationships between the classes of nodes in two different networks are modeled as graph

regularization constraints. Unlike our previous binary class formulation [4], it also allows us to incorporate prior knowledge about the potential relationships between classes in different networks to avoid overfitting. Second, we applied our framework to co-classify Wikipedia users and articles and showed that it significantly outperforms classifiers that learn each classification task independently.

The remainder of this paper is organized as follows. Section II briefly reviews the related work on Wikipedia and other network classification problems. Section III summarizes the notations and terminology used in the paper. In Section IV, we present the methodology for our proposed co-classification framework. Section V presents the experimental results. Finally, conclusions and future work are described in Section VI.

II. Related Work

Classification of articles or users is an important task that underlies many interesting research problems in Wikipedia. One example is vandalism detection, where the objective is to revert changes made by vandals with the intent of degrading the quality of information on a Wikipedia page. For example, Smets et al. [14] developed an autonomous system to distinguish vandalism from legitimate edits in Wikipedia using a naïve bayes classifier and a probabilistic sequence modeling approach. Druck et al. [7] proposed a probabilistic model to detect the quality of edits in Wikipedia. Potthast et al. [11] constructed various features to facilitate the detection of vandalism. In addition to vandalism, there have been some previous works on measuring the quality of Wikipedia articles. For example, Zeng et al. [18] used a dynamic Bayesian network to compute the trust value of an article based on its revision history. Hu et al. [10] developed three quality measures for Wikipedia articles, taking into consideration interaction data between the articles and their editors. In addition to vandalism detection, other potential applications of Wikipedia classification include bot detection, spam detection, topic detection, and automated categorization of user interests.

Wikipedia classification can be considered as a special case of network classification problem, in which the objective is to assign each node in a network to one of the k possible class labels. Most of the current techniques employ both the node features and link information to aid the classification task [20] [9][21]. For example, Zhu et al. [21] developed a matrix factorization algorithm that exploits both the content and linkage information to address the Web page classification problem. There have also been quite extensive works on using graph regularization methods to combine link structure with content information. For example, Zhang et al. [20] developed a risk minimization formulation for learning from both text and graph structures. In [13], Lu and Getoor designed a structured logistic regression model to capture both content and link information. Gan and Suel [12] proposed a two-stage approach to improve a classifier's performance using the link structure. While all of these existing methods attempt to combine link structure and node features to enhance performance of network classification, none of them were designed to solve multiple related classification tasks in different networks. We consider them as single-network, single-classification tasks.

We had recently developed a co-classification framework for detecting web spam and spammers simultaneously in a social bookmarking Web site [4]. Their approach was inspired by the fact that both Web spam and spammer detection tasks are intuitively related. Thus, it would be advantageous to train their classifiers simultaneously to leverage the labeled examples available on both users and articles. The experimental results is inspiring. However, the method proposed in [4] considers only a binary class problem. In this paper, we extended the formulation to deal with multi-class problem and enables the incorporation of a prior matrix as a bias to the algorithm.

III. PRELIMINARIES

We begin with a brief discussion of the terminology and notations used in this paper. The terminology is given in the context of user and article networks in Wikipedia.

- **User:** A registered user is identified by a username and has an associated user page on Wikipedia. Unregistered users are also allowed to edit the Wikipedia pages. These users are identified by their corresponding IP address. In this study, we consider only registered Wikipedia users. Let U be the set of such users.
- **Article:** Articles in Wikipedia have a title and are organized by categories. Some articles are considered Wikipedia stubs, which has been created but yet to contain any useful information. In this study, we consider only non-stub Wikipedia articles. Let A be the set of such articles.
- **User-Article Matrix:** The edits made by users on Wikipedia articles are represented in a user-article matrix M, where $M(i, j)$ is the total number of edits made by user i on article j.
- **Category relation Matrix:** Let E_t be a category relation matrix estimated from the training data. The matrix represents the number of links between each user category and article category. For instance, $E_t(y^{(u)}, y^{(a)})$ is the total number of edits made by users from category $y^{(u)}$ on articles from category $y^{(a)}$, where $y^{(u)} \in \{1, 2, ...k_u\}$ and $y^{(a)} \in \{1, 2, ..., k_a\}$. k_u and k_a are the number of user and article categories, respectively.

Our proposed framework assumes that there is a correspondence between user and article categories (classes). For example, suppose we are interested in classifying the topic areas of Wikipedia users and articles. We expect users who are classified as "computer science" editors to edit more articles classified as "computer science" than other areas

such as "biology". As will be shown in Section IV, this assumption can be enforced using a graph regularization constraint in our proposed co-classification framework.

Before describing our methodology, we first formalize the co-classification problem. Suppose we are given:

- A set of l_u labeled users, $\mathcal{L}^{(u)} = \{(x_1^{(u)}, y_1^{(u)}), (x_2^{(u)}, y_2^{(u)}), \cdots, (x_{l_u}^{(u)}, y_{l_u}^{(u)})\}$ where $x_i^{(u)}$is a d_u-dimensional feature vector for the i-th user and $y_i^{(u)} \in \{1, 2...k_u\}$ is its class label.
- A set of $n_u - l_u$ unlabeled users, $\mathcal{U}^{(u)} = \{x_{l_u+1}^{(u)}, x_{l_u+2}^{(u)}, \cdots, x_{n_u}^{(u)}\}$
- A set of l_a labeled articles, $\mathcal{L}^{(a)} = \{(x_1^{(a)}, y_1^{(a)}), (x_2^{(a)}, y_2^{(a)}), ...(x_{l_a}^{(a)}, y_{l_a}^{(a)})\}$ where $x_j^{(a)}$is a d_a-dimensional feature vector for the j-th article and $y_j^{(a)} \in \{1, 2, \cdots, k_a\}$ is its class label.
- A set of $n_a - l_a$ unlabeled articles, $\mathcal{U}^{(a)} = \{x_{l_a+1}^{(a)}, x_{l_a+2}^{(a)}, \cdots, x_{n_a}^{(a)}\}$
- A set of user-article pairs, $\mathcal{M}$, where $(x_i^{(u)}, x_j^{(a)}) \in \mathcal{M}$ if user $x_i^{(u)}$ edits article $x_j^{(a)}$.

***Definition** 1:* (Co-Classification). Given $\mathcal{L}^{(u)}, \mathcal{L}^{(a)}$, and $\mathcal{M}$, the goal of co-classification is to learn a pair of classifiers: (1) $f_u : \Re^{d_u} \rightarrow \{1, 2, \cdots, k_u\}$ that accurately maps the feature vector of a user $x^{(u)}$ to its class label $y^{(u)}$ and (2) $f_a : \Re^{d_a} \rightarrow \{1, 2, \cdots, k_a\}$ that accurately maps the feature vector of an article $x^{(a)}$ to its class label $y^{(a)}$.

IV. METHODOLOGY

In this section, we first introduce a k-class support vector machine and a least squares support vector machine (LS-SVM). We then present our proposed co-classification framework, which is an extension of the LS-SVM classifier.

A. Support Vector Machine

A support vector machine (SVM) is a classifier trained to construct an optimal hyperplane with a maximal margin that separates examples from different classes [6]. It can be applied to classify data sets with nonlinear decision surfaces using the kernel trick. The solution to the problem of finding an optimal separating hyperplane can be formulated as a quadratic programming problem involving inequality constraints. SVM can be applied to a k-class problem by transforming it into multiple binary classification tasks. The most common strategies include 1-vs-rest and 1-vs-1 [3][16]. The former train k independent binary SVM classifiers, one for each class, where the j^{th} classifier $y_j(x)$ is trained using the labeled instances from the j-th class as positive examples and instances from the remaining classes as the negative examples. The class label of a new input x is predicted as follows:

$$y(x) = \max_k y_k(x) \tag{1}$$

Another strategy for k-class SVM is 1-vs-1, which trains $k(k-1)/2$ binary SVM classifiers on all possible pairs of classes. A test point is then classified based on the class that has the highest number of votes.

B. Least Squares Support Vector Machines

One potential limitation of regular SVM is the high computational cost in solving the quadratic programming problem. To alleviate this problem, Suykens et al. [15] proposed a least-square support vector machine (LS-SVM) formulation, which involves solving an objective function with quadratic term but with equality constraints. The solution to the objective function is obtained by solving a system of linear equations. Previous studies have found that the generalization performance of LS-SVM is comparable to that of regular SVM [8][17]. Furthermore, when the data points are linearly independent, it has been theoretically shown that LS-SVM is equivalent to hard margin SVM using the Mahalanobis distance measure [19].

The framework developed in this paper is an extension of the LS-SVM formulation for network data developed in [4]. Before describing the framework, we first consider the task of classifying Wikipedia users and articles independently. Given a set of labeled users $\mathcal{L}^{(u)}$, we may construct a LS-SVM classifier for users by minimizing the following objective function:

$$\phi(w, e) = \frac{1}{2}\sum_{m=1}^{k_u}(w_m^{(u)T} w_m^{(u)}) + \gamma_1 \frac{1}{2}\sum_{i=1}^{l_u}\sum_{m \neq y_i} e_{im}^{(u)2} \tag{2}$$

subject to the following constraints

$$w_{y_i}^{(u)} x_i^{(u)} + b_{y_i}^{(u)} = w_m^{(u)} x_i^{(u)} + b_m^{(u)} + 2 - e_{im}^{(u)}$$

$$e_{im}^{(u)} \geq 0,\ i = 1, 2, ..., l_u,\ m \in \{1, 2, ...k_u\} \backslash y_i$$

where $\{w_m^{(u)}\}$ is the set of parameters for the user classifier, γ_1 is a parameter that controls the penalty of misclassifying users, and $e_{im}^{(u)}$ is the error of classifying the i-th user. The constraint optimization problem can be cast into the following problem using the Lagrangian formulation:

$$L_u(w, b, e, \alpha) = \frac{1}{2}\sum_{m=1}^{k_u}(w_m^{(u)T} w_m^{(u)}) + \gamma_1 \frac{1}{2}\sum_{i=1}^{l_u}\sum_{m \neq y_i} e_{im}^{(u)2}$$

$$- \sum_{i=1}^{l_u}\sum_{m \neq y_i} \alpha_{im}(w_{y_i}^{(u)} x_i^{(u)} + b_{y_i}^{(u)} - w_m x_i^{(u)} - b_m^{(u)} - 2 + e_{im}^{(u)}) \tag{3}$$

where $\{\alpha_{im}\}$ is the set of Lagrange multipliers. Similarly, the corresponding Lagrangian formulation for classifying articles can be expressed as follow:

$$L_a(w, b, \varepsilon, \beta) = \frac{1}{2}\sum_{n=1}^{k_a}(w_n^{(a)T} w_n^{(a)}) + \gamma_2 \frac{1}{2}\sum_{j=1}^{l_a}\sum_{n \neq y_j} \varepsilon_{jn}^{(a)2}$$

$$-\sum_{j=1}^{l_a}\sum_{n\neq y_j}\beta_{jn}(w_{y_j}^{(a)}x_j^{(a)}+b_{y_j}^{(a)}-w_n^{(a)}x_j^{(a)}-b_n^{(a)}-2+\varepsilon_{jn}^{(a)}) \tag{4}$$

where $\{w_n^{(a)}\}$ is the set of parameters for the article classifier, γ_2 is a parameter that controls the penalty of misclassifying articles, and $\varepsilon_{jn}^{(a)}$ is the error of classifying the j-th article.

C. Co-Classification of Users and Articles

In this paper, instead of solving the optimization problems for classifying users and articles independently, we propose a co-classification algorithm that utilizes the link structure between users and articles from the two relation matrices E_t and M described in Section III. The category relation matrix E_t is computed as follow:

$$E_t(y^{(u)},y^{(a)})=\sum_{\substack{i:y_i^{(u)}=y^{(u)}\\ j:y_j^{(a)}=y^{(a)}}}M(x_i^{(u)},x_j^{(a)})$$

where $x_i^{(u)}\in\mathcal{L}^{(u)}$, $y^{(u)}\in\{1,2,...k_u\}$, $x_j^{(a)}\in\mathcal{L}^{(a)}$, and $y^{(a)}\in\{1,2,...k_a\}$.

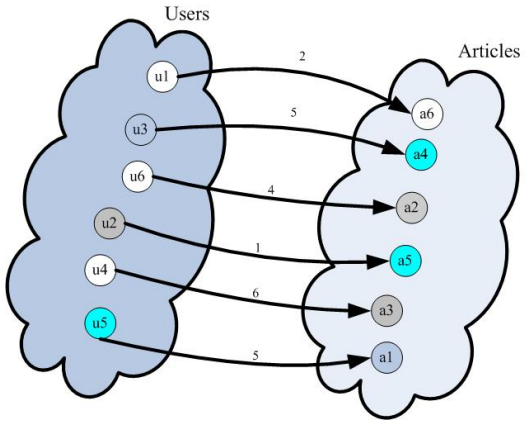

Figure 1. A Sample of User-Article network

Figure 1 shows an example of the links between Wikipedia users and articles. Suppose both the users and articles are divided into four categories, represented by nodes with the following colors: white, blue, gray, and cyan. The edge between the user and article corresponds to the number of edits, which is captured by the user-article matrix M. For example, $M(u_6,a_2)=4$ means user u_6 have edited the article a_2 four times. Each entry in the category relation matrix $E_t(y^{(u)},y^{(a)})$ corresponds to the sum of the weights of edges between users from category $y^{(u)}$ and articles from category $y^{(a)}$. For example, E_t(white, gray) $=M(u_6,a_2)+M(u_4,a_3)=10$.

One limitation of using E_t is that if the number of labeled users connected to the labeled articles is small, then the relationship between the user and article classes may not be accurately represented by E_t. Furthermore, the presence of noisy links in Wikipedia data also have an adverse affect that may degrade classification performance. In many cases, we may have some prior knowledge about the probability of a link between nodes in two classes in the user and article networks. This information can be encoded using a prior matrix E_p to avoid overfitting the model to the observed relationships between classes in the different networks. Based on these two considerations, we construct an adjusted category relation matrix as follows:

$$E(y^{(u)},y^{(a)})=\alpha E_t(y^{(u)},y^{(a)})+(1-\alpha)E_p(y^{(u)},y^{(a)}), \tag{5}$$

where $0\leq\alpha\leq 1$ is a tuning parameter that controls the tradeoff between E_t and E_p. Its value can be determined via cross-validation or set to a constant, say 0.5, for equal weighting. We use the following prior matrix in our experiments: where the rows and columns are the user and article

Table I
PRIOR CATEGORY RELATION MATRIX $E_p(k_u,k_a)$

	C_{a1}	C_{a2}	C_{a3}	C_{a4}
C_{u1}	1	-1	-1	-1
C_{u2}	-1	1	-1	-1
C_{u3}	-1	-1	1	-1
C_{u4}	-1	-1	-1	1

categories, respectively.

The values in E_t should be normalized for two reasons. First, the values in E_t and E_p do not have the same scale. The larger values in E_t tend to dominate the overall value of E, thus losing information about our prior knowledge. Second, it is useful to normalize the values in E_t so that it ranges between [-1,1]. As will be seen shortly, this helps to enforce a penalty or reward scheme for our regularization constraint. Based on these two considerations, we normalize E_t as follows:

$$E_t^{''}(y_i^{(u)},y_j^{(a)})=E_t(y_i^{(u)},y_j^{(a)})/\sum_{k_a}E_t(y_i^{(u)},y_j^{(a)})$$

$$E_t^{'}(y_i^{(u)},y_j^{(a)})=E_t^{''}(y_i^{(u)},y_j^{(a)})-\frac{\sum_{k_a}E_t^{''}(y_i^{(u)},y_j^{(a)})}{k_a}$$

After normalization, if $y_i^{(u)}$ and $y_j^{(a)}$ are related categories, then $E_t^{'}(y_i^{(u)},y_j^{(a)})$ is positive-valued; otherwise, $E_t^{'}(y_i^{(u)},y_j^{(a)})$ is negative-valued. The normalized value for $E_t^{'}$ will be used to estimate E in (5).

The category relation matrix will be used to enforce a constraint between the user and article classifiers. Specifically, the assumption described in Section III states that Wikipedia users will not likely edit articles in their non-related categories. The following graph regularization constraint is therefore introduced to penalize models in which

users edit articles that belong to non-related categories:

$$-\gamma_3 \sum_{i=1}^{l_u}\sum_{j=1}^{l_a} \sum_{M(x_i^{(u)},x_j^{(a)})\neq 0} E^{'}(y_i^{(u)}, y_j^{(a)})(w_{y_i^{(u)}} x_i^{(u)} + b_{y_i^{(u)}} - w_{y_j^{(a)}} x_j^{(a)} - b_{y_j^{(a)}})$$

The overall objective function for our co-classification framework of users and articles can be written as follows:

$$\begin{aligned} L &= \frac{1}{2}\sum_{m=1}^{k_u}(w_m^{(u)T} w_m^{(u)}) + \gamma_1 \frac{1}{2}\sum_{i=1}^{l_u}\sum_{m\neq y_i} e_{im}^{(u)2} \\ &+ \frac{1}{2}\sum_{n=1}^{k_a}(w_n^{(a)T} w_n^{(a)}) + \gamma_2 \frac{1}{2}\sum_{j=1}^{l_a}\sum_{n\neq y_j} \xi_{jn}^{(a)2} \\ &- \sum_{i=1}^{l_u}\sum_{m\neq y_i} \alpha_{im}(w_{y_i}^{(u)} x_i^{(u)} + b_{y_i}^{(u)} - w_m^{(u)} x_i^{(u)} - b_m^{(u)} - 2 + e_{im}^{(u)}) \\ &- \sum_{j=1}^{l_a}\sum_{n\neq y_j} \beta_{jn}(w_{y_j}^{(a)} x_j^{(a)} + b_{y_j}^{(a)} - w_n^{(a)} x_j^{(a)} - b_n^{(a)} - 2 + \varepsilon_{jn}^{(a)}) \\ &- \gamma_3 \sum_{i,j} \sum_{M(x_i^{(u)},x_j^{(a)})\neq 0} E^{'}(y_i^{(u)}, y_j^{(a)})(w_{y_i}^{(u)} x_i^{(u)} + b_{y_i}^{(u)} \\ &- w_{y_j}^{(a)} x_j^{(a)} - b_{y_j}^{(a)}) \end{aligned}$$

where $(w^{(u)}, w^{(a)}, e, \xi, \alpha, \beta)$ are the model parameters to be estimated from training data and $(\gamma_1, \gamma_2, \gamma_3)$ are user-specified parameters. For our experiments, we set $\gamma_1 = \gamma_2 = 1$ whereas γ_3 is estimated from the data via cross validation.

The objective function is solved by taking the derivative of L with respect to each of the model parameters and setting them to zero. This leads to the following set of linear equations.

$$\frac{\partial L}{\partial w_m^{(u)}} = 0 \Rightarrow$$

$$w_m^{(u)} + \sum_{i=1}^{l_u}\sum_{y_i^{(u)}\neq m} \alpha_{im} x_i^{(u)} - \sum_{i=1}^{l_u}\sum_{y_i^{(u)}=m, l\neq m} \alpha_{il} x_i^{(u)} - \gamma_3 \sum_{i,j}\sum_{\substack{y_i^{(u)}=m \\ M(x_i^{(u)},x_j^{(a)})\neq 0}} E^{'}(m, y_j^{(a)}) x_i^{(u)} = 0$$

$$\frac{\partial L}{\partial b_m^{(u)}} = 0 \Rightarrow$$

$$\gamma_3 \sum_{i,j}\sum_{\substack{y_i^{(u)}=m \\ M(x_i^{(u)},x_j^{(a)})\neq 0}} E^{'}(m, y_j^{(a)}) + \sum_{i=1}^{l_u}\sum_{\substack{l\neq m \\ y_i^{(u)}=m}} \alpha_{il} = \sum_{i=1}^{l_u}\sum_{y_i^{(u)}\neq m} \alpha_{im}$$

$$\frac{\partial L}{\partial e_{im}^{(u)}} = 0 \Rightarrow \gamma_1 e_{im}^{(u)} I(y_i^{(u)} \neq m) = \alpha_{im} I(y_i^{(u)} \neq m)$$

$$\frac{\partial L}{\partial \alpha_{im}} = 0 \Rightarrow I(y_i^{(u)} \neq m)(w_{y_i}^{(u)} x_i^{(u)} + b_{y_i}^{(u)} - w_m^{(u)} x_i^{(u)} - b_m^{(u)} - 2 + e_{im}^{(u)}) = 0$$

$$\frac{\partial L}{\partial w_n^{(a)}} = 0 \Rightarrow$$

$$w_n^{(a)} + \sum_{j=1}^{l_a}\sum_{y_j^{(a)}\neq n} \alpha_{jn} x_j^{(a)} - \sum_{j=1}^{l_a}\sum_{y_j^{(a)}=n, p\neq n} \alpha_{ip} x_j^{(a)} + \gamma_3 \sum_{i,j}\sum_{\substack{y_j^{(a)}=n \\ M(x_i^{(u)},x_j^{(a)})\neq 0}} E^{'}(y_i^{(u)}, n) x_j^{(a)} = 0$$

$$\frac{\partial L}{\partial b_n^{(a)}} = 0 \Rightarrow$$

$$-\gamma_3 \sum_{i,j}\sum_{\substack{y_j^{(a)}=n \\ M(x_i^{(u)},x_j^{(a)})\neq 0}} E^{'}(y_i^{(u)}, n) + \sum_{j=1}^{l_a}\sum_{\substack{y_j^{(a)}=n \\ p\neq n}} \beta_{ip} = \sum_{j=1}^{l_a}\sum_{y_j^{(a)}\neq n} \beta_{jn} \quad (6)$$

$$\frac{\partial L}{\partial \xi_{jn}^{(a)}} = 0 \Rightarrow \gamma_2 \xi_{jn}^{(a)} I(y_j^{(a)} \neq n) = \beta_{jn} I(y_j^{(a)} \neq n)$$

$$\frac{\partial L}{\partial \beta_{jn}} = 0 \Rightarrow I(y_j^{(a)} \neq n)(w_{y_j}^{(a)} x_j^{(a)} + b_{y_j}^{(a)} - w_n^{(a)} x_j^{(a)} - b_n^{(a)} - 2 + \xi_{jn}^{(a)}) = 0$$

in which I is an indicate function,

$$I(a,b) = \begin{cases} 1 & a \neq b \\ 0 & a = b \end{cases}$$

The preceding set of linear equations can be expressed in matrix notation as follows:

$$\begin{bmatrix} \mathbf{C^{(u)}} & \mathbf{0} \\ \mathbf{0} & \mathbf{C^{(a)}} \end{bmatrix} \begin{bmatrix} \chi^{(\mathbf{u})} \\ \chi^{(\mathbf{a})} \end{bmatrix} = \begin{bmatrix} \mathbf{b^{(u)}} \\ \mathbf{b^{(a)}} \end{bmatrix} \quad (7)$$

where $C^{(u)}$ is a matrix that contains $x_i^{(u)}$ and $E^{'}(m, y_j^{(a)})$, the vector $\chi^{(u)} = (w_1^{(u)}, ..., w_{k_u}^{(u)}, b_1^{(u)}, ..., b_{k_u}^{(u)}, e, \alpha)^T$, the vector $b^{(u)} = (p_1, q_1, 0_{l_u \times k_u}, 2_{l_u \times k_u})^T$, $0_{l_u \times k_u}$ is a $l_u \times k_u$-dimensional vector of 0s, $2_{l_u \times k_u}$ is a $l_u \times k_u$-dimensional vector of 2s. Analogously, $C^{(a)}$, $\chi^{(a)}$, and

$b^{(a)} = (p_2, q_2, 0_{l_a \times k_a}, 2_{l_a \times k_a})^T$ are the corresponding matrix and vectors for solving the parameters of the article classifier, where:

$$\mathbf{p_1(m)} = \gamma_3 \sum_{i,j} \sum_{\substack{y_i^{(u)}=m \\ M(x_i^{(u)}, x_j^{(a)})\neq 0}} E^{'}(m, y_j^{(a)}) \mathbf{x_i^{(u)}}$$

$$q_1(m) = \gamma_3 \sum_{i,j} \sum_{\substack{y_i^{(u)}=m \\ M(x_i^{(u)}, x_j^{(a)})\neq 0}} E^{'}(m, y_j^{(a)})$$

$$\mathbf{p_2(n)} = \gamma_3 \sum_{i,j} \sum_{\substack{y_j^{(a)}=n \\ M(x_i^{(u)}, x_j^{(a)})\neq 0}} E^{'}(y_i^{(u)}, n) \mathbf{x_j^{(a)}}$$

$$q_2(n) = \gamma_3 \sum_{i,j} \sum_{\substack{y_j^{(a)}=n \\ M(x_i^{(u)}, x_j^{(a)})\neq 0}} E^{'}(y_i^{(u)}, n)$$

The block structure of the matrix equation in Equation (7) suggests that the system of linear equations can be decoupled into two subproblems, one for learning the parameters of the user classifier and the other for article classifier.

$$\mathbf{C^{(u)}} \chi^{(\mathbf{u})} = \mathbf{b^{(u)}} \tag{8}$$

$$\mathbf{C^{(a)}} \chi^{(\mathbf{a})} = \mathbf{b^{(a)}} \tag{9}$$

It is worth noting that the solutions for both equations are not entirely independent because the terms p_1, p_2, q_1 and q_2 depend on the link structure of M and E. It is sufficient to solve Equations (8) and (9) for $(\alpha, b_m^{(u)}, \beta, b_n^{(a)})$ in order to classify the unlabeled users and articles. Specifically, a user $x_{test}^{(u)}$ and an article $x_{test}^{(a)}$ can be classified as follows:

$$\begin{aligned} f_u(x_{test}^{(u)}) &= \max_m (w_m^{(u)} x_{test}^{(u)} + b_m^{(u)}) \\ &= \max_m \Big(\sum_{i=1}^{l_u} \sum_{\substack{y_i^{(u)}=m \\ l\neq m}} \alpha_{il} x_i^{(u)T} x_{test}^{(u)} - \sum_{i=1}^{l_u} \sum_{y_i^{(u)}\neq m} \alpha_{im} x_i^{(u)T} x_{test}^{(u)} \\ &\quad + \gamma_3 \sum_{i,j} \sum_{\substack{y_i^{(u)}=m \\ M(x_i^{(u)}, x_j^{(a)})\neq 0}} E^{'}(m, y_j^{(a)}) x_i^{(u)T} x_{test}^{(u)} + b_m^{(u)} \Big) \end{aligned}$$

$$\begin{aligned} f_a(x_{test}^{(a)}) &= \max_n (w_n^{(a)} x_{test}^{(a)} + b_n^{(a)}) \\ &= \max_n \Big(\sum_{j=1}^{l_a} \sum_{\substack{y_j^{(a)}=n \\ p\neq n}} \alpha_{ip} x_j^{(a)T} x_{test}^{(a)} - \sum_{j=1}^{l_a} \sum_{y_j^{(a)}\neq n} \alpha_{jn} x_j^{(a)T} x_{test}^{(a)} \\ &\quad + \gamma_3 \sum_{i,j} \sum_{\substack{y_j^{(a)}=n \\ M(x_i^{(u)}, x_j^{(a)})\neq 0}} E^{'}(y_i^{(u)}, n) x_j^{(a)T} x_{test}^{(a)} + b_n^{(a)} \Big) \end{aligned}$$

Algorithm 1 summarizes the high-level overview of our co-classification algorithm. The `linearsolver` function solves the set of linear equations to obtain the model parameters. The `classify` function takes the models parameters and unlabeled data as input to generate their predicted class labels as output.

`Algorithm 1` Co-Classification Algorithm:
`Input:` $\mathcal{U}^{(a)}$, $\mathcal{U}^{(u)}$, $\mathcal{L}^{(a)}$,$\mathcal{L}^{(u)}$, M, $E^{'}$ and γ
`Output:` $(y_i^{(u)}|i = l_u + 1, ...n_u)$, $(y_j^{(a)}|j = l_a + 1, ...n_a)$
`Method:`
1. $(\alpha, \beta, b^{(u)}, b^{(a)}) \leftarrow$ linearsolver($\mathcal{L}^{(a)}, \mathcal{L}^{(u)}$, M, $E^{'}$, γ)
2. $y^{(a)} \leftarrow$ Classify($\mathcal{U}^{(a)}, \alpha, b^{(a)}$)
3. $y^{(u)} \leftarrow$ Classify($\mathcal{U}^{(u)}, \beta, b^{(u)}$)

V. EXPERIMENTAL EVALUATION

A. Data Set

To evaluate the performance of our algorithm, we downloaded the Wikipedia dump dated Oct-09-2009. After preprocessing, we chose the following four topics from each network as our ground truth classes ($k_u = k_a = 4$): biology, natural science, computer science and political science. The label for each user is assigned based on label for the majority class of articles written by the user. We created a sample that contains 5361 users and 6403 articles for our experiments.

The experimental results reported in this section is obtained using 5-fold cross validation. For each run, we use 4 of the folds for training and the remaining fold as test data. We repeated the experiment ten times, each time using a random division of the data into five folds. Table II shows the category relation matrix $E_t^{'}$ obtained from the training data.

Table II
CATEGORY RELATION MATRIX FROM TRAINING DATA $E_t^{'}$

	C_{a1}	C_{a2}	C_{a3}	C_{a4}
C_{u1}	0.6658	-0.2225	-0.2274	-0.2159
C_{u2}	-0.2186	0.6983	-0.2385	-0.2412
C_{u3}	-0.2342	-0.2271	0.5430	-0.0816
C_{u4}	-0.2423	-0.2462	-0.2398	0.7283

B. Evaluation Metrics

The performance of the classification algorithms are evaluated using the following metrics:

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

$$Recall = \frac{TP}{TP + FN} \tag{11}$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{12}$$

Table III
FOUR CATEGORIES AVERAGE RESULTS

	Article			User		
5-fold cross validation	*Pre*	*Rec*	F_1	*Pre*	*Rec*	F_1
LS-SVM	48.8169	49.6420	49.2260	41.5724	40.6410	41.1014
Co-LS-SVM	50.1656	53.5612	51.8078	42.8800	46.2185	44.4867

where TP is the number of positive example correctly predicted by the classification model. FP is the number of negative examples wrongly predicted as positive by the classification model. FN is the number of positive examples wrongly predicted as negative by the classification model.

Precision means the percentage of positive examples predicted correctly among all examples classified as positive class, while recall measures the fraction of all positive examples detected by the classifier. We use the well-known F_1 measure to summarize the precision and recall into a single value. F_1 is the harmonic mean between precision and recall.

C. Experimental Results and Discussion

Our experimental results is summarized in Table III. The table shows a comparison between the performance of our proposed multi-class LS-SVM framework against applying the LS-SVM algorithm independently on the user and article networks. For the latter case, we employ the 1-vs-rest strategy to train the independent LS-SVM models. Note that *Pre* means the average precision for all four categories, *Rec* means the average recall for all four categories, and F_1 is the harmonic mean of precision and recall. Clearly, the F_1 measure for our proposed framework is higher than that for LS-SVM.

A more detailed analysis can be seen in Figures 2 and Figure 3, where we have shown the F_1 values for each class in the article and user networks. Notice that the improvement in F_1 is observed across all the classes. These results suggest that our multi-class graph regularization framework along with the addition of prior matrix enables our algorithm to outperform LS-SVM models that are trained independently.

Despite the observed improvement, our overall F_1 measure is around 50%, which is not very high. This probably is because the presence of noisy links in Wikipedia that adversely affects our classification results. Furthermore, we use the content of the user page in Wikipedia to represent the feature vector of each user. Since some of the user pages contain only a few uninformative words or even empty, their feature vector may not be effectively used for classification. As a next step in our work, we may consider removing the noisy links and exploring additional features to represent the users.

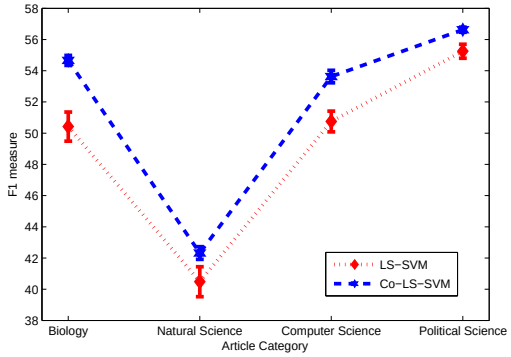

Figure 2. Comparison of F_1 measure for each Article category between LS-SVM and Co-LS-SVM algorithms

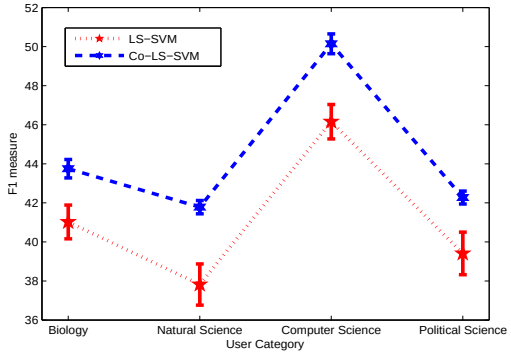

Figure 3. Comparison of F_1 measure for each User category between LS-SVM and Co-LS-SVM algorithms

VI. CONCLUSION AND FUTURE WORK

This paper introduces a novel multi-class co-classification model for Wikipedia user and article networks. We formalized the joint classification tasks as a constraint optimization problem, in which the relationships between articles and users who edit these articles are captured using a graph regularization term. We demonstrated that our framework is more effective than LS-SVM classifiers that learns each classification task independently.

For future work, we plan to incorporate article-article and user-user networks into our co-classification framework, similar to the approach presented in our earlier work. Specifically, we may add the following two assumptions:

- **Article-Article Assumption:** Articles that are linked together are expected to have the same class.
- **User-User Assumption:** Users that are linked together are expected to have the same class.

We may also introduce more subcategories in Wikipedia

data to test the effectiveness of our algorithm. For example, each of the four topics we used in this paper can be further divided into subtopics: political science = {civil-rights liberties; imperialism; nationalism; political theories}; natural science = {physics; earth sciences; chemistry; astronomy}; computer science = {algorithms; operating systems; data bases}; computer architecture; artificial intelligence}; biology = {genetics; zoology; anatomy; cell-biology; neuroscience}. Finally, we plan to extend the formulation to a semi-supervised co-classification setting.

REFERENCES

[1] J. Abernethy, O. Chapelle, and C. Castillo. Web spam identification through content and hyperlinks. In *Proceeding Of The Sigir Workshop On Adverasrial Information Retrieval On Web (Airweb'08)*, Beijing,China, 2008.

[2] G. Attardi, A. Gull, and F. Sebastiani. Automatic web page categorization by link and context analysis, 1999.

[3] C. M. Bishop. *Pattern Recognition And Machine Learning*. Springer Verlag, 2006.

[4] F. Chen, P. Tan, and A. Jain. A co-classification framework for detecting web spam and spammers in social media web sites. In *Proceeding Of ACM CIKM International Conference On Information And Knowledge Management*, Hong Kong,China, 2009.

[5] D. Cohn and T. Hofmann. The missing link - a probabilistic model of document content and hypertext connectivity, 2001.

[6] C. CORTES and V. VAPNIK. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

[7] G. Druck, G. Miklau, and A. McCallum. Learning to predict the quality of contributions to wikipedia. In *Proceedings of the AAAI Workshop on Wikipedia and Artificial Intelligence (WIKIAI 08)*, pages 7–12, 2008.

[8] T. Gestel, J. Suykens, J. V. B. Baesens, S. Viaene, G. Dedene, B. Moor, and J. Vandewalle. Benchmarking least square support vector machine classifiers. *Machine Learning*, 54(1):5–32, 2004.

[9] R. Heatherly, M. Kantarcioglu, and B. Thuraisingham. Social Network Classification Incorporating Link Type. In *IEEE Intelligence and Security Informatics*, , Dallas, Texas,, June 2009.

[10] M. Hu, E.-P. Lim, A. Sun, H. W. Lauw, and B.-Q. Vuong. Measuring article quality in wikipedia models and evaluation. 2007.

[11] M. Potthast, B. Stein, and R. Gerling. Automatic vandalism detection in wikipedia. *Advances in Information Retrieval*, pages 663–668, 2008.

[12] Q.Gan and T.Suel. Improving web spam classifiers using link structure. In *Proceeding of the SIGIR Workshop on Adversarial Information Retrieval on the Web(AIRWEB'07)*, Banff,Canada, 2007.

[13] Q.Lu and L.Getoor. Link-based classification. In *Proceeding of the twentieth International Conference on Machine Learning(ICML2003)*, Washion DC, 2003.

[14] K. Smets, B. Goethals, and B. Verdonk. Automatic vandalism detection in wikipedia: Towards a machine learning approach. In *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI)*, pages 43–48. AAAI Press, 2008.

[15] J. Suykens, T. Gestel, J. Brabanter, B. Moor, and J. Vandewalle. *Least Squares Support Vector Machines*. World Scientific, Singapore, 2002.

[16] J. Weston and C. Watkins. Support vector machines for multi-class pattern recognition, 1999.

[17] J. Ye and T. Xiong. Svm versus least square svm. In *Proceeding Of The Eleventh International Conference On Artificial Intelligence And Statistics*, Puerto Rico, 2007.

[18] H. Zeng, M. A.Alhossaini, L. Ding, R. Fikes, and D. L.McGuinness. Computing trust from revision history. In *International Conference on Privacy, Security and Trust*, 2006.

[19] P. Zhang and J. Peng. Svm vs regularized least squares classification. In *17th International Conference on Pattern Recognition(ICPR'04)*, UK, 2004.

[20] T. Zhang, A. Popescul, and B. Dom. Linear prediction models with graph regularization for web-page categorization. In *Proceeding of ACM SIGKDD International Conf on Data Mining*, pages 812–826, Philadelphia,PA, 2006.

[21] S. Zhu, K. Yu, Y. Chi, and Y. Gong. Combining content and link for classification using matrix factorization. In *Proceeding Of ACM SIGIR*, Netherlands, 2007.