Activity-edge Centric Multi-label Classification for Mining Heterogeneous Information Networks

Yang Zhou
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332
yzhou@gatech.edu

Ling Liu
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332
lingliu@cc.gatech.edu

ABSTRACT

Multi-label classification of heterogeneous information networks has received renewed attention in social network analysis. In this paper, we present an activity-edge centric multi-label classification framework for analyzing heterogeneous information networks with three unique features. First, we model a heterogeneous information network in terms of a collaboration graph and multiple associated activity graphs. We introduce a novel concept of vertex-edge homophily in terms of both vertex labels and edge labels and transform a general collaboration graph into an activity-based collaboration multigraph by augmenting its edges with class labels from each activity graph through activity-based edge classification. Second, we utilize the label vicinity to capture the pairwise vertex closeness based on the labeling on the activity-based collaboration multigraph. We incorporate both the structure affinity and the label vicinity into a unified classifier to speed up the classification convergence. Third, we design an iterative learning algorithm, AEClass, to dynamically refine the classification result by continuously adjusting the weights on different activity-based edge classification schemes from multiple activity graphs, while constantly learning the contribution of the structure affinity and the label vicinity in the unified classifier. Extensive evaluation on real datasets demonstrates that AEClass outperforms existing representative methods in terms of both effectiveness and efficiency.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Data Mining

Keywords

Multi-label Classification; Heterogeneous Network; Activity-based Edge Classification; Collaboration Multigraph; Label Vicinity

1. INTRODUCTION

Multi-label classification has received increasing attention in both data mining and machine learning over the last decade [10–22]. In contrast to single-label classification, multi-label classification analysis adopts a more realistic view that entities in the real world are often associated with multiple class labels simultaneously. For example, most people in a social network belong to multiple social groups and participate in multiple types of activities with different degrees of engagement. Most of web pages in the web graph may cover multiple topics at different intensities.

Existing multi-label classification efforts for networked data focus on designing effective and yet scalable algorithms [17–22]. Although previous studies differ from one another in the concrete approaches to mining the linkage structure, to the best of our knowledge, they all suffer from two weaknesses: (1) None of previous studies separate different types of activity graphs from the heterogeneous information networks and exploit the correlations among the set of class labels within each activity graph and across multiple activity graphs; and (2) None of previous works combine both the vertex-centric multi-label classification and the edge-centric multi-label classification to boost the effectiveness and efficiency.

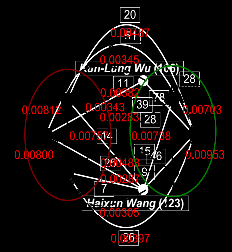
In this paper we show that by utilizing activity-edge centric approach, we can incorporate the two missing dimensions to improve both the accuracy and the complexity of multi-label classification analysis. First, we argue that entities in the real world may involve themselves in multiple activity networks. These activity networks may provide abundant information about heterogeneous entities and links, and how entities are linked in the context of each of activity networks. We aim to utilize these activity networks to find a natural and cheap way to identify the inter-dependencies among labels. Second, based on different activity networks, an entity can be tagged by a subset of K labels with different class-membership distributions. We model the class-membership distribution for each of activity networks as multi-labeled edges. Third, we consider not only the labels of related vertices but also the possible labels of associated edges to further enhance the accuracy of multi-label classification. We integrate the vertex-centric labeling and edge-centric labeling into a unified classifier with different weights. An iterative method is proposed to learn the weights towards the classification objective.

This paper makes the following original contributions to multi-label classification for networked data.

- We model a heterogeneous information network in terms of a collaboration graph and multiple associated activity graphs, and cluster activity vertices in each activity graph into K categories with the given K class labels. Clustering each activity graph provides a natural way to capture the dependencies among activity categories within activity graphs.
- We introduce a novel concept of vertex-edge homophily in terms of both vertex labels and edge labels, and transform a general collaboration graph into an activity-based collaboration multigraph by augmenting its edges with class labels from each activity graph through activity-based edge classification.
- We utilize the structure affinity to capture the pairwise topological similarity of vertices and the label vicinity to capture the pairwise vertex closeness based on the labeling on the activity-based collaboration multigraph. We incorporate both the struc-

KDD'14, August 24–27, 2014, New York, NY, USA.

http://dx.doi.org/10.1145/2623330.2623737.


20
0.00345
28
11
0.00382
0.00812
0.00343
39
0.00703
0.00283
28
0.00738
0.00800
0.00953
0.00287
7
0.00305

$y_i = (y_i^1, y_i^2, \cdots, y_i^K) \in \{0,1\}^K$, in which $y_i^j = 1$ iff the label c_j is in the label set of v_i. We use $Y = \{y_1, \cdots, y_l, y_{l+1}, \cdots, y_{N_{CG}}\}$ to denote a possible labeling for the instance set V. $Y_l = \{y_1, \cdots, y_l\}$ indicates the observed multi-label set assigned to V_l and $Y_u = Y - Y_l$ represents the multi-label set to be determined. The task of our **activity-edge centric multi-label classification of multigraph** is to use the label information of the training instances in V_l to predict the label set Y_u for the testing instances in V_u.

3. THE AEClass APPROACH

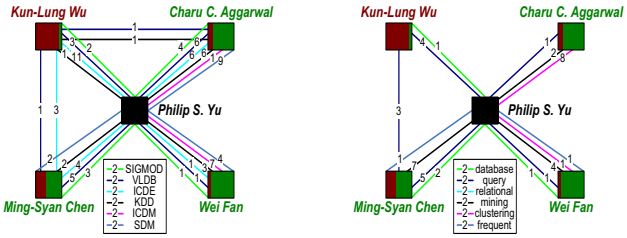
Compared to existing multi-label relational classifiers outlined in Section 1, AEClass improves both the accuracy and the efficiency of multi-label classification by incorporating four mining strategies: (1) activity-based edge classification; (2) edge label dependency; (3) vertex label vicinity; and (4) weight learning. We first introduce the overall design of AEClass. We then describe each part of AEClass in detail in the next subsections.

- Activity-based edge classification, which consists of five tasks. (1) given a collaboration graph CG, choose N suitable activity graphs AG_i based on the specific context defined by the classification objective; (2) cluster all AG_is into K activity categories; (3) construct an label dependency graph based on the clustering of each AG_i to identify inter-dependencies among K class labels; (4) based on K categories of each AG_i, split and classify each unlabeled edge in CG into at most K labeled edges; and (5) transform CG and all AG_is into a unified multigraph MG by integrating N edge classification schemes of CG based on each AG_i weighted by $\omega_1^{(1)}=\cdots=\omega_N^{(1)}=\frac{1}{N}$.
- Activity-edge centric vertex classification, which includes four tasks. (1) initialize a transition probability $\mathbf{T}_j^{(1)}$ of MG; (2) initialize a classification kernel $\mathbf{K}_j^{(1)}$; (3) infer the class-membership vector $\mathbf{X}_j^{(1)}$ on each class c_j; and (4) produce the class-membership vector $\mathbf{Y}_j^{(1)}$ by refining $\mathbf{X}_j^{(1)}$ with label dependency graphs.
- Iterative learning, which has four steps. (1) solve the parametric programming problem for classification objective to update $\alpha^{(t)}, \beta^{(t)}$, $\omega_1^{(t)}, ..., \omega_N^{(t)}$ ($\alpha^{(t)}, \beta^{(t)}$ are updated if t>2); (2) adjust the structure affinity $\mathbf{T}_j^{(t+1)}$ of CG with $\omega_1^{(t)}, ..., \omega_N^{(t)}$; (3) update $\mathbf{K}_j^{(t+1)}$ by combining the structure affinity $\mathbf{T}_j^{(t+1)}$ and the label vicinity $(\mathbf{T}_j^{(t+1)}\mathbf{Y}^{(t)})(\mathbf{T}_j^{(t+1)}\mathbf{Y}^{(t)})^T$ weighted by $\alpha^{(t)}$ and $\beta^{(t)}$ ($\alpha^{(t)}=\beta^{(t)}=\frac{1}{2}$ if t=2); and (4) do classification $\mathbf{X}_j^{(t+1)}=\mathbf{K}_j^{(t+1)}\mathbf{X}_j^{(t)}$ and enter the next round.

3.1 Activity-based Edge Classification

Existing classification models assume the existence of vertex homophily, namely, similar vertices in nature are connected to each other with social links. For example, *Philip S. Yu* and *Wei Fan* have many co-authored works published on *DM* conferences, as shown in Figure 4 (a). However, the truth is not always like this. entities that are connected together may be similar in different ways with respect to a given set of K class labels. As is known to all, *Philip S. Yu* and *Ming-Syan Chen* are experts on data mining, i.e., they both have more research publications in the area of data mining than in any other academic area such as database. However, as seen in Figure 4 (a), they have more co-authored papers published on *DB* conferences. Thus the vertex homophily is insufficient to accurately infer the possible labels of an author. This motivates us to propose the concept of vertex-edge homophily, the principle that both links and their associated vertices should be similar and likely belong to the same classes, to further improve the accuracy of multi-label classification.

In order to capture the vertex-edge homophily in the multi-label classification, we first perform activity-based edge classification.



(a) Conference-based Splitting (b) Term-based Splitting

Figure 3: Edge Splitting

For each activity graph and the original collaboration graph CG, we first construct an activity-edge augmented collaboration graph CG_i by examining each edge and the pair of connected entities in CG and splitting each edge into a set of parallel edges based on each activity in AG_i that this pair of entities have in common. The size of each set of parallel edges is at most N_{AG_i}, i.e., the number of activity vertices in AG_i. Figures 3 (a) and (b) present the activity-edge augmented collaboration graphs of Figure 1 (a) based on conference activities in Figure 1 (b) and term activities in Figure (c) respectively. Each edge in Figure 1 (a) is divided into multiple edges in terms of the common conference venues or the common title terms in co-authored publications between the pair of co-authors.

However, such activity-based edge augmentation may lead to substantial increase in size of activity-edge augmented collaboration graphs. We address this issue by introducing activity-based edge augmentation with edge classification to efficiently improve the scalability of classification. Concretely, we utilize the clustering result by NetClus, i.e., the probability distribution of each activity over K categories, to infer the class labels of parallel edges in each CG_i over the K categories.

Given the probability of the m^{th} activity in AG_i belonging to cluster (class) c_j produced by NetClus, denoted by $P(L_m = c_j|AG_i)$, we can compute the class-membership probability of edge $(v_p, v_q) \in E$ belonging to class c_j based on AG_i, denoted by $P(L_{pq} = c_j|AG_i)$.

$$P(L_{pq} = c_j|AG_i) = \frac{1}{W(p,q)} \sum_{m=1}^{N_{AG_i}} W_m^i(p,q) P(L_m = c_j|AG_i) \quad (1)$$

where $W(p,q)$ represents the value on edge $(v_p, v_q) \in E$ in CG, and $W_m^i(p,q)$ denotes the value on the m^{th} edge between v_p and v_q in CG_i, which is based on the m^{th} activity in AG_i. If there does not exist such an edge between v_p and v_q, then $W_m^i(p,q)$ is equal to 0.

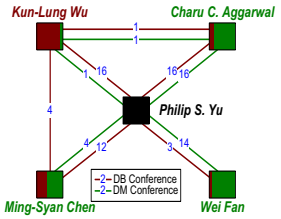
After generating the class-membership distribution of each edge in CG_i, we reduce CG_i to an activity-edge augmented collaboration graph $\overline{CG_i}$ with classified edges by grouping at most N_{AG_i} parallel edges between any pair of vertices in CG_i into at most K parallel edges in $\overline{CG_i}$.

$$\overline{W}_j^i(p,q) = W(p,q) P(L_{pq} = c_j|AG_i) \quad (2)$$

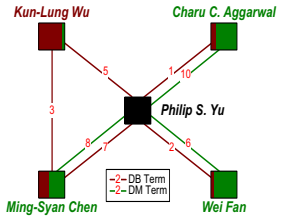
where $\overline{W}_j^i(p,q)$ represents the value on the edge with label c_j between v_p and v_q in $\overline{CG_i}$. For ease of presentation, assuming that *SIGMOD*, *VLDB*, *ICDE*, *database*, *query* and *relational* only belong to class *DB* with the probability of 1, and *KDD*, *ICDM*, *SDM*, *mining*, *clustering* and *frequent* just belong to class *DM* with the probability of 1, two $\overline{CG_i}$s in Figure 4 present the edge-classification results of two CG_is in Figures 3 respectively.

3.2 Activity-edge Centric Vertex Classification

As N edge classification schemes of CG, i.e., $\overline{CG_i}$s ($1 \le i \le N$), may have different degree of contributions to vertex classification, we propose to integrate N edge classification schemes into a unified collaboration multigraph with different weighting factors $\omega_1^{(t)}, \cdots, \omega_N^{(t)}$ through dynamic weight tuning mechanism. Thus the unified weight value on the edge with label c_j between v_p and v_q in MG at the t^{th} iteration, denoted by $\mathbf{W}_j^{(t)}(p,q)$, can be computed as follow.



(a) Conference-based Classification



(b) Term-based Classification

Figure 4: Edge Classification

$$\mathbf{W}_j^{(t)}(p,q) = \sum_{i=1}^{N} \omega_i^{(t)} \overline{W}_j^i(p,q) = \sum_{i=1}^{N} \omega_i^{(t)} \sum_{m=1}^{N_{AG_i}} W_m^i(p,q) P(L_m = c_j | AG_i),\ 1 \le j \le K \tag{3}$$

subject to $\sum_{i=1}^{N} \omega_i^{(t)} = 1$, $\omega_i^{(t)} \geqslant 0$, $i = 1, \cdots, N$.

Note that $\mathbf{W}_j^{(t)}(p,q)$ keeps changing with $\omega_1^{(t)}, \cdots, \omega_N^{(t)}$ through dynamic weight learning. We set the initial $\mathbf{W}_j^{(1)}(p,q)$ with equal weighting factors of $\omega_1^{(1)}, \cdots, \omega_N^{(1)} = \frac{1}{N}$.

Figure 5 (a) shows the unified multigraph for our running example in Figure 1 by combining the links with the same labels between the same vertex pair from two activity-based edge classification schemes in Figure 4 with equal weighting factor of 0.5.

With the unified multigraph MG, we below describe the activity-edge centric vertex classification, which integrates the activity-edge labels with the vertex labels among structurally relevant instances through transition probability on collaboration multigraph.

Definition 1. [Transition Probability on Collaboration Multigraph] Let $MG = (V, E)$ be a *collaboration multigraph* where V is the set of entity vertices and E is the set of parallel edges denoting the collaborative relationships on different classes between entities of MG. The transition probability on MG at the t^{th} iteration can be defined by normalizing the edge values as follows.

$$\mathbf{T}_j^{(t)}(p,q) = \begin{cases} \dfrac{\mathbf{W}_j^{(t)}(p,q)}{\sum_{r=1}^{N_{CG}} \sum_{m=1}^{K} \mathbf{W}_m^{(t)}(p,r)}, & p > l, \\ 1, & p = q \le l, \\ 0, & otherwise. \end{cases} \tag{4}$$

where $\mathbf{T}_j^{(t)}(p,q)$ represents the transition probability on the edge with label c_j between v_p and v_q in MG. Here, we assume that Y_l, i.e., the labels of the vertices in V_l, are fixed during the classification process. Figure 5 (b) presents the transition probabilities of parallel edges from *Philip S. Yu* to other authors based on the collaboration multigraph in Figure 5 (a).

We express the above transition probability in a matrix form.

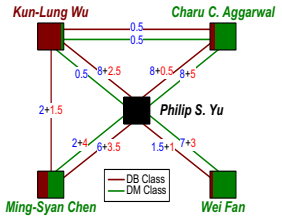
$$\mathbf{T}_j^{(t)} = (\mathbf{D}^{(t)})^{-1} \mathbf{W}_j^{(t)} \tag{5}$$

where $\mathbf{D}^{(t)}$ is a diagonal matrix $\mathbf{D}^{(t)} = \text{diag}(1, \cdots, 1, d_{l+1}, \cdots, d_{N_{CG}})$, $1, \cdots, 1$ specifies l ones, and $d_p = \sum_{r=1}^{N_{CG}} \sum_{m=1}^{K} \mathbf{W}_m^{(t)}(p,r)$ $(l+1 \le p \le N_{CG})$. $\mathbf{T}_j^{(t)}$ determines the transition probability on those edges with the class label of c_j in MG.

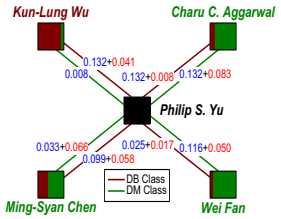
Instead of decomposing the multi-label classification problem into to a set of binary classification problems, we construct a unified multi-label classifier by using a single normalizing factor $\mathbf{D}^{(t)}$ to normalize parallel edges with different class labels. The original transition operation is actually divided into two steps: (1) choose those edges with the objective class label in terms of classification objective; and (2) select an edge with the largest value from the above edges to jump.

Now we define the initial unified classification kernel $\mathbf{K}_j^{(1)}$, which only utilizes the structure information of MG, i.e., those edges with label c_j, due to the lack of label vicinity at initialization.

$$\mathbf{K}_j^{(1)} = \mathbf{T}_j^{(1)} \tag{6}$$



(a) Coauthor Multigraph



(b) Transition Probability

Figure 5: Multigraph Representation

Since we have ordered the vertices in V such that the labeled nodes V_l are indexed before the unlabeled nodes V_u, we rewrite the unified classification kernel $\mathbf{K}_j^{(1)}$ as a block matrix.

$$\mathbf{K}_j^{(1)} = \begin{bmatrix} \mathbf{K}_{jll}^{(1)} & \mathbf{K}_{jlu}^{(1)} \\ \mathbf{K}_{jul}^{(1)} & \mathbf{K}_{juu}^{(1)} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{O} \\ \mathbf{K}_{jul}^{(1)} & \mathbf{K}_{juu}^{(1)} \end{bmatrix} \tag{7}$$

where $\mathbf{K}_{jll}^{(1)}$ is an $l \times l$ identity matrix representing the transition probability among labeled vertices, we set the $l \times (N_{CG} - l)$ block matrix $\mathbf{K}_{jlu}^{(1)}$ to be zero matrix since the labels on the vertices in V_l are fixed, the $(N_{CG} - l) \times l$ matrix $\mathbf{K}_{jul}^{(1)}$ specifies the transition probability from unlabeled vertices to labeled vertices, and $\mathbf{K}_{juu}^{(1)}$ is an $(N_{CG} - l) \times (N_{CG} - l)$ matrix denoting the transition probability among unlabeled vertices.

Suppose that the class-membership matrix is denoted by $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_K] \in \mathbb{R}^{N_{CG} \times K}$, for each class-membership vector $\mathbf{X}_j (1 \le j \le K)$ based on class c_j, we use its individual classification kernel $\mathbf{K}_j^{(t)}$ to iteratively infer the probabilities of vertices on class c_j.

$$\mathbf{X}_j^{(t)} = \mathbf{K}_j^{(t)} \mathbf{X}_j^{(t-1)} \tag{8}$$

Let $\mathbf{X}_j = [\mathbf{X}_{jl}; \mathbf{X}_{ju}]$ be the class-membership vector, where $\mathbf{X}_{jl}$ indicates the probabilities of the labeled vertices in V_l belonging to class c_j, and $\mathbf{X}_{ju}$ represents the probabilities of the unlabeled vertices in V_u belonging to class c_j. Due to the labels on the vertices in V_l are fixed, Eq. (8) is equivalent to the following formula.

$$\mathbf{X}_{ju}^{(t)} = \mathbf{K}_{jul}^{(t)} \mathbf{X}_{jl}^{(t-1)} + \mathbf{K}_{juu}^{(t)} \mathbf{X}_{ju}^{(t-1)} \tag{9}$$

After the t^{th} iteration, the class-membership matrix is updated as follow.

$$\mathbf{X}^{(t)} = [\mathbf{X}_1^{(t)}, \mathbf{X}_2^{(t)}, \cdots, \mathbf{X}_K^{(t)}] = \begin{bmatrix} \mathbf{X}_{1l}^{(t)} & \mathbf{X}_{2l}^{(t)} & \cdots & \mathbf{X}_{Kl}^{(t)} \\ \mathbf{X}_{1u}^{(t)} & \mathbf{X}_{2u}^{(t)} & \cdots & \mathbf{X}_{Ku}^{(t)} \end{bmatrix} \tag{10}$$

Compared to existing multi-label relational classifiers, we argue that AEClass based on the activity-edge augmented collaboration multigraph can significantly improve the performance of multi-label classification: (1) accuracy improvement. Based on the vertex-edge homophily, we classify each edge in CG into at most K parallel edges in MG. During the classification process, AEClass only picks up those vertices and links with the same label as the current objective class c_j, i.e., $\mathbf{K}_j^{(t)}$ and $\mathbf{X}_j^{(t-1)}$, to execute the inference. For example, given the class-membership probabilities of *Ming-Syan Chen* on classes *DB* and *DM* in Figure 5 (a), we want to infer the class-membership probabilities of *Kun-Lung Wu* on *DB* and *DM*. AEClass will produce a positive probability on *DB* and a zero probability on *DM* since there exists no edge with label *DM* between these two authors. In contrast, existing classifiers will output a higher probability on *DM* than on *DB* for any positive edge value between two authors in the original CG in Figure 1 (a). In fact, *Kun-Lung Wu* is known as a database researcher without any data mining publications. (2) efficiency improvement. Based on the vertex homophily, no matter which class the current objective is, existing classifiers need to check each neighbor of a vertex and summarize the labels of all neighbors. In comparison, AEClass performs the similar summary at lower cost. When we classify an edge in CG into m parallel edges in MG, m is often much smaller than the number of K class labels. Suppose that the current objective

(a) Conference Dependency (b) Term Dependency

Figure 6: Edge Label Dependency by Category Similarity

class is c_j, for a neighbor of a vertex, there may not exist an edge with label c_j between the vertex and this neighbor. Thus, the number of neighbors of a vertex with edge label c_j can be much smaller than the number of its neighbors, thus reducing the amount of unnecessary computations. Concretely, by utilizing the vertex-edge homophily, AEClass only needs to consider those links with label c_j and associated neighbors and further stops label propagation to the circle of those irrelevant neighbors (without link with label c_j) in the next iterations. For the above example, we can safely ignore the operation of inferring the probability of *Kun-Lung Wu* on *D-M* since there exists no edge with label *DM* between two authors. More importantly, AEClass prevents the probability of *Ming-Syan Chen* on *DM* from being diffused to both *Kun-Lung Wu* and the neighbor-based circle of *Kun-Lung Wu*.

3.3 Improvement by Edge Label Dependency

We argue that the underlying correlations among different activity categories can have significant impact on the performance of multi-label classification. Based on activity graph partition, we first define the edge label similarity to capture the inter-dependencies among K activity categories within each of N activity graphs.

Definition 2. [Edge Label Similarity] Let $AG_i = (U_i, F_i)$ be the i^{th} activity graph ($1 \le i \le N$), $S_i(u_m, u_n)$ be the similarity score between two activities $u_m, u_n \in U_i$ in AG_i, and U_{ip} and U_{iq} be two categories of U_i with class labels of $c_p, c_q \in C$ respectively. The activity category similarity between c_p and c_q with respect to AG_i is also referred to as the edge label dependency between two edge labels c_p and c_q, and is defined as follow.

$$S_i(c_p, c_q) = \begin{cases} \sum_{u_m \in U_{ip}, u_n \in U_{iq}} \frac{S_i(u_m, u_n)}{|U_{ip}| \times |U_{iq}|}, & p \neq q, \\ 1, & p = q. \end{cases} \tag{11}$$

Figure 6 shows two edge label dependency graph by activity category similarity based on two class labels *DB* and *DM* with respect to the conference graph and the term graph in Figure 2 respectively.

We thus incorporate them into our AEClass framework to adjust the class-membership matrix $\mathbf{X}^{(t)}$. The adjusted class-membership vector on class c_j, denoted by $\mathbf{Y}'^{(t)}_{ju}$, can be defined by integrating class-membership vectors on other classes in terms of the similarity scores between class c_j and other classes.

$$\mathbf{Y}'^{(t)}_{ju} = \sum_{m=1}^{K} \sum_{i=1}^{N} \omega_i^{(t)} S_i(c_j, c_m) \mathbf{X}^{(t)}_{mu}, \ 1 \le j \le K \tag{12}$$

where the weight $\omega_i^{(t)}$ for AG_i is the same as in Eq. (3). The adjusted class-membership matrix is thus defined as follow.

$$\mathbf{Y}'^{(t)} = [\mathbf{Y}'^{(t)}_1, \mathbf{Y}'^{(t)}_2, \cdots, \mathbf{Y}'^{(t)}_K] = \begin{bmatrix} \mathbf{X}_{1l} & \mathbf{X}_{2l} & \cdots & \mathbf{X}_{Kl} \\ \mathbf{Y}'^{(t)}_{1u} & \mathbf{Y}'^{(t)}_{2u} & \cdots & \mathbf{Y}'^{(t)}_{Ku} \end{bmatrix} \tag{13}$$

3.4 Refinement by Vertex Label Vicinity

One disadvantage of conventional iterative classifiers is that they often need lots of iterations to converge to a stationary distribution and the repeated label propagation causes a non-trivial computational cost. Wang et al. [22] proposed a dynamic label propagation (DLP) model by fusing both data features and data labels to improve the effectiveness on multi-class/multi-label classification. However, the DLP model failed to quantify the weighted contributions from data features and data labels such that it often can not work well on real classification tasks. We model the label vicinity to capture the pairwise vertex closeness based on the labeling on the activity-based collaboration multigraph by following the similar idea. To improve both effectiveness and efficiency of classification, we design an iterative learning method to dynamically refine the classification results by continuously quantifying and adjusting the weights on the structure affinity and on the label vicinity towards the classification objective.

Based on the transition probability $\mathbf{T}_j^{(t)}$ on CG, we define a diffusion process to map the multigraph space into an N_{CG}-dimensional space $\mathbb{R}^{N_{CG}}$, where each element $\phi_j^{(t)}(i) \in \mathbb{R}^{N_{CG}}$ represents the transition probabilities on the edges with label c_j from vertex v_i to the other vertices and $P(\phi_j^{(t)}(i)) = \mathcal{N}(\phi_j^{(t)}(i)|\mathbf{T}_j^{(t)})$. On the other hand, based on a heuristics rule: two instance vertices with highly similar class-membership distributions are likely to be highly similar to each other in the input multigraph space, $\mathbf{Y}^{(t)}(\mathbf{Y}^{(t)})^T$ can be viewed as the similarity between vertices based on the class-membership distribution. Similarly, we map this label-based similarity space into an N_{CG}-dimensional space $\mathbb{S}^{N_{CG}}$, where each entry $\varphi^{(t)}(i) \in \mathbb{S}^{N_{CG}}$ specifies the label-based similarity between vertex v_i and the other vertices and $P(\varphi^{(t)}(i)) = \mathcal{N}(\varphi^{(t)}(i)|\mathbf{Y}^{(t)}(\mathbf{Y}^{(t)})^T)$. We then define a linear projection operation based on $\mathbf{T}_j^{(t+1)}$.

$$\phi_j^{(t+1)} = \mathbf{T}_j^{(t+1)} \varphi^{(t)} \tag{14}$$

where $\phi_j^{(t+1)} = [\phi_j^{(t+1)}(1); \phi_j^{(t+1)}(2); \cdots; \phi_j^{(t+1)}(N_{CG})]$ and $\varphi^{(t)} = [\varphi^{(t)}(1); \varphi^{(t)}(2); \cdots; \varphi^{(t)}(N_{CG})]$.

With the linear projection, we generate the following formula.

$$P(\phi_j^{(t+1)}|\varphi^{(t)}) = \mathcal{N}(\phi_j^{(t+1)}|\mathbf{T}_j^{(t+1)}\varphi^{(t)}) \tag{15}$$

The corresponding marginal distribution is given below.

$$\begin{aligned} P(\phi_j^{(t+1)}) &= \int \mathcal{N}(\varphi^{(t)}|\mathbf{Y}^{(t)}(\mathbf{Y}^{(t)})^T)\mathcal{N}(\phi_j^{(t+1)}|\mathbf{T}_j^{(t+1)}\varphi^{(t)})d\varphi^{(t)} \\ &= \mathcal{N}(\phi_j^{(t+1)}|\mathbf{T}_j^{(t+1)}\mathbf{Y}^{(t)}(\mathbf{Y}^{(t)})^T(\mathbf{T}_j^{(t+1)})^T) \end{aligned} \tag{16}$$

Since directly combining $\varphi^{(t)}(\varphi^{(t)})^T$ into the unified classification kernel $\mathbf{K}_j^{(t+1)}$ may lead to a degeneration at the beginning of classification if the learned label information of vertices in V_u is not enough to infer the label-based similarity scores, we adjust $\mathbf{K}_j^{(t+1)}$ by integrating the label-based similarity through $\mathbf{T}_j^{(t)}$.

$$\mathbf{K}_j^{(t+1)} = \alpha^{(t+1)}\mathbf{T}_j^{(t+1)} + \beta^{(t+1)}(\mathbf{T}_j^{(t+1)}\mathbf{Y}^{(t)})(\mathbf{T}_j^{(t+1)}\mathbf{Y}^{(t)})^T \tag{17}$$

subject to $\alpha^{(t+1)} + \beta^{(t+1)} = 1, \alpha^{(t+1)}, \beta^{(t+1)} \ge 0$.

$\alpha^{(t+1)}$ and $\beta^{(t+1)}$ are weighting factors to balance two kinds of similarity scores. The label vicinity $(\mathbf{T}_j^{(t+1)}\mathbf{Y}^{(t)})(\mathbf{T}_j^{(t+1)}\mathbf{Y}^{(t)})^T$ quantitatively measures the extent of similarity between vertices and their neighbors based on the current labeling.

3.5 Weight Learning

Classification analysis often utilizes the F1 score, i.e., the harmonic mean of precision and recall, to evaluate the accuracy of testing instances. The objective of multi-label classification of multigraph is to maximize the Macro-F1 score [32], i.e., the unweighted mean of F1 score on classes. To define the Macro-F1 score, we first introduce an indicator function.

$$\mathcal{I}(\hat{y}_i^j = 1) = \begin{cases} 1, & \hat{y}_i^j = 1, \\ 0, & \hat{y}_i^j = 0. \end{cases} \tag{18}$$

where $\mathcal{I}(\hat{y}_i^j = 1)$ indicates whether the label c_j is assigned to an instance vertex v_i.

Definition 3. [Macro-F1] Let $MG = (V, E)$ be a collaboration multigraph, y_i be the true label vector of the i^{th} instance vertices in V and $\hat{y}_i$ be the predicted label vector, and the Macro-F1 score is defined below.

$$Macro\text{–}F1 = \frac{1}{K} \sum_{j=1}^{K} \frac{2 \sum_{i=l+1}^{N_{CG}} \mathcal{I}(\hat{y}_i^j = 1) y_i^j}{\sum_{i=l+1}^{N_{CG}} \mathcal{I}(\hat{y}_i^j = 1) + \sum_{i=l+1}^{N_{CG}} y_i^j} \tag{19}$$

Assuming $\theta = \max_{i,j}\{\mathbf{Y}(i,j) : l+1 \le i \le N_{CG}, 1 \le j \le K\}$, we define an s-shape function to approximate the indicator function.

$$\mathcal{S}(\mathbf{Y}(i,j)) = \begin{cases} 1, & \mathbf{Y}(i,j) = \theta, \\ 0, & \mathbf{Y}(i,j) = 0, \\ \mathbf{Y}(i,j)/\theta, & otherwise. \end{cases} \tag{20}$$

The decision rule determining $\hat{y}_i^j = 1$ if $\mathbf{Y}(i,j) > \theta/2$, i.e., $\mathcal{S}(\mathbf{Y}(i,j)) > 0.5$ is represented as follow.

$$\mathcal{I}(\hat{y}_i^j = 1) = \mathcal{I}(\mathbf{Y}(i,j) > \theta/2) = \mathcal{I}(\mathcal{S}(\mathbf{Y}(i,j)) > 0.5) \approx \mathcal{S}(\mathbf{Y}(i,j)) \tag{21}$$

The Macro-F1 score is thus approximated as follow.

$$Macro\text{–}F1 \approx \frac{1}{K}\sum_{j=1}^{K}\frac{2\sum_{i=l+1}^{N_{CG}}\mathbf{Y}(i,j)y_i^j}{\sum_{i=l+1}^{N_{CG}}\mathbf{Y}(i,j)+\sum_{i=l+1}^{N_{CG}}\theta y_i^j} \tag{22}$$

According to Eqs.(3)-(17), the Macro-F1 score is a fractional function of multi variables $\alpha, \beta, \omega_1, \cdots, \omega_N$ with non-negative real coefficients. On the other hand, the numerator and the denominator of Macro-F1 are both polynomial functions of the above variables. Without loss of generality, we rewrite Eq.(22) as follow.

$$Macro\text{–}F1 \approx \frac{\sum_{i=1}^{m} a_i(\alpha)^{b_i}(\beta)^{c_i}\prod_{j=1}^{N}(\omega_j)^{d_{ij}}}{\sum_{i=1}^{n} o_i(\alpha)^{p_i}(\beta)^{q_i}\prod_{j=1}^{N}(\omega_j)^{r_{ij}}}, \tag{23}$$

$$a_i, b_i, c_i, d_{ij}, o_i, p_i, q_i, r_{ij} \geq 0, b_i, c_i, d_{ij}, p_i, q_i, r_{ij} \in \mathbb{Z}$$

where there are m polynomial terms in the numerator and n polynomial terms in the denominator, a_i and o_i are the coefficients of the i^{th} terms respectively, and $b_i, c_i, d_{ij}, p_i, q_i, r_{ij}$ are the exponents of corresponding variables in the i^{th} terms respectively.

Definition 4. [Multigraph Classification Objective] Let $MG = (V, E)$ be a collaboration multigraph, $\alpha, \beta, \omega_1, \cdots, \omega_N$ are the weighting factors defined in Eqs.(3) and (17), respectively. The goal of multi-label classification of multigraph is to maximize the Macro-F1 score.

$$\max_{\alpha,\beta,\omega_j} Macro\text{–}F1 \approx \max_{\alpha,\beta,\omega_j} \frac{\sum_{i=1}^{m} a_i(\alpha)^{b_i}(\beta)^{c_i}\prod_{j=1}^{N}(\omega_j)^{d_{ij}}}{\sum_{i=1}^{n} o_i(\alpha)^{p_i}(\beta)^{q_i}\prod_{j=1}^{N}(\omega_j)^{r_{ij}}} \tag{24}$$

subject to $\alpha + \beta = 1, \alpha, \beta \geqslant 0, \sum_{j=1}^{N}\omega_j = 1, \omega_j \geqslant 0, j = 1, \cdots, N$.

For ease of presentation, we revise the original objective as the following nonlinear fractional programming problem (NFPP).

Definition 5. [Nonlinear Fractional Programming Problem] Let $f(\alpha, \beta, \omega_1, \ldots, \omega_N) = \sum_{i=1}^{m} a_i(\alpha)^{b_i}(\beta)^{c_i}\prod_{j=1}^{N}(\omega_j)^{d_{ij}}$ and $g(\alpha, \beta, \omega_1, \ldots, \omega_N) = \sum_{i=1}^{n} o_i(\alpha)^{p_i}(\beta)^{q_i}\prod_{j=1}^{N}(\omega_j)^{r_{ij}}$, the classification goal is revised below.

$$\max_{\alpha,\beta,\omega_1,\ldots,\omega_N} \frac{f(\alpha,\beta,\omega_1,\ldots,\omega_N)}{g(\alpha,\beta,\omega_1,\ldots,\omega_N)} \tag{25}$$

subject to $\alpha + \beta = 1, \alpha, \beta \geqslant 0, \sum_{i=1}^{N}\omega_i = 1, \omega_i \geqslant 0, i = 1, \cdots, N$.

Our classification objective is equivalent to maximize a quotient of two polynomial functions of multiple variables. It is very hard to perform function trend identification and estimation to determine the existence and uniqueness of solutions. Therefore, we want to transform this sophisticated NFPP into a easily solvable problem.

Theorem 1. *The NFPP in Definition 5 is equivalent to a polynomial programming problem with polynomial constraints (PPPPC).*

$$\max_{\alpha,\beta,\omega_1,\ldots,\omega_N,\pi} \pi f(\alpha,\beta,\omega_1,\ldots,\omega_N) \tag{26}$$

subject to $\alpha + \beta = 1, \alpha, \beta \geqslant 0, \sum_{i=1}^{N}\omega_i = 1, \omega_i \geqslant 0, i = 1, \cdots, N, 0 \leqslant \pi \leqslant 1/g(\alpha, \beta, \omega_1, \ldots, \omega_N)$.

Proof. If $(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N, \overline{\pi})$ *is an optimal solution of PPPPC, then* $\overline{\pi} = 1/g(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)$. *Thus* $\overline{\pi} f(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N) = f(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)/g(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)$. *For any feasible solution* $(\alpha, \beta, \omega_1, \ldots, \omega_N)$ *of NFPP, the constraints of PPPPC are satisfied by setting* $\pi = 1/g(\alpha, \beta, \omega_1, \ldots, \omega_N)$, *so* $\pi f(\alpha, \beta, \omega_1, \ldots, \omega_N) \leqslant \overline{\pi} f(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)$, *i.e.* $f(\alpha, \beta, \omega_1, \ldots, \omega_N)/g(\alpha, \beta, \omega_1, \ldots, \omega_N) \leqslant f(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)/g(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)$.

Conversely, if $(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)$ *solves NFPP, then for any feasible solution* $(\alpha, \beta, \omega_1, \ldots, \omega_N, \pi)$ *of PPPPC we have* $\pi f(\alpha, \beta, \omega_1, \ldots, \omega_N) \leqslant f(\alpha, \beta, \omega_1, \ldots, \omega_N)/g(\alpha, \beta, \omega_1, \ldots, \omega_N) \leqslant f(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)/g(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N) = \overline{\pi} f(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)$ *with* $\overline{\pi} = 1/g(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)$.

Although PPPPC is a polynomial programming problem, the polynomial constraints make it very hard to solve. We further simplify it as an nonlinear parametric programming problem (NPPP).

Definition 6. [Nonlinear Parametric Programming Problem] Let $f(\alpha, \beta, \omega_1, \ldots, \omega_N) = \sum_{i=1}^{m} a_i(\alpha)^{b_i}(\beta)^{c_i}\prod_{j=1}^{N}(\omega_j)^{d_{ij}}$ and $g(\alpha, \beta, \omega_1, \ldots, \omega_N) = \sum_{i=1}^{n} o_i(\alpha)^{p_i}(\beta)^{q_i}\prod_{j=1}^{N}(\omega_j)^{r_{ij}}$, the NPPP is defined below.

$$F(\gamma) = \max_{\alpha,\beta,\omega_1,\ldots,\omega_N} f(\alpha,\beta,\omega_1,\ldots,\omega_N) - \gamma g(\alpha,\beta,\omega_1,\ldots,\omega_N) \tag{27}$$

subject to $\alpha + \beta = 1, \alpha, \beta \geqslant 0, \sum_{i=1}^{N}\omega_i = 1, \omega_i \geqslant 0, i = 1, \cdots, N$.

Theorem 2. *The NFPP in Definition 5 is equivalent to the NPPP in Definition 6, i.e.,* $\gamma = \max_{\alpha,\beta,\omega_1,\ldots,\omega_N} \frac{f(\alpha,\beta,\omega_1,\ldots,\omega_N)}{g(\alpha,\beta,\omega_1,\ldots,\omega_N)}$ *iff* $F(\gamma) = \max_{\alpha,\beta,\omega_1,\ldots,\omega_N} f(\alpha,\beta,\omega_1,\ldots,\omega_N) - \gamma g(\alpha,\beta,\omega_1,\ldots,\omega_N) = 0$.

Proof. If $(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)$ *is a feasible solution of* $F(\gamma) = 0$, *then* $f(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N) - \gamma g(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N) = 0$. *Thus* $f(\alpha, \omega_1, \ldots, \omega_N) - \gamma g(\alpha, \omega_1, \ldots, \omega_N) \leqslant f(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N) - \gamma g(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N) = 0$. *We have* $\gamma = f(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)/g(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N) \geqslant f(\alpha, \omega_1, \ldots, \omega_N)/g(\alpha, \omega_1, \ldots, \omega_N)$. *Therefore,* γ *is a maximum value of NFPP and* $(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)$ *is an optimal solution of NFPP.*

Conversely, if $(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)$ *solves NFPP, then we have* $\gamma = f(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)/g(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N) \geqslant f(\alpha, \omega_1, \ldots, \omega_N)/g(\alpha, \omega_1, \ldots, \omega_N)$. *Thus* $f(\alpha, \omega_1, \ldots, \omega_N) - \gamma g(\alpha, \omega_1, \ldots, \omega_N) \leqslant f(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N) - \gamma g(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N) = 0$. *We have* $F(\gamma) = 0$ *and the maximum is taken at* $(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)$.

Now the original NFPP has been successfully transformed into the straightforward NPPP. This transformation can efficiently speed up the classification convergence due to the following properties.

Theorem 3. $F(\gamma)$ *is convex.*

Proof: Suppose that $(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)$ *is an optimal solution of* $F((1-\lambda)\gamma_1 + \lambda\gamma_2)$ *with* $\gamma_1 \neq \gamma_2$ *and* $0 \leqslant \lambda \leqslant 1$. $F((1-\lambda)\gamma_1 + \lambda\gamma_2) = f(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N) - ((1-\lambda)\gamma_1 + \lambda\gamma_2)g(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N) = \lambda(f(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N) - \gamma_2 g(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)) + (1-\lambda)(f(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N) - \gamma_1 g(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)) \leqslant \lambda \max_{\alpha,\beta,\omega_1,\ldots,\omega_N} f(\alpha, \beta, \omega_1, \ldots, \omega_N) - \gamma_2 g(\alpha, \beta, \omega_1, \ldots, \omega_N) + (1-\lambda) \max_{\alpha,\beta,\omega_1,\ldots,\omega_N} f(\alpha, \beta, \omega_1, \ldots, \omega_N) - \gamma_1 g(\alpha, \beta, \omega_1, \ldots, \omega_N) = \lambda F(\gamma_2) + (1-\lambda)F(\gamma_1)$. *Thus,* $F(\gamma)$ *is convex.*

Theorem 4. $F(\gamma)$ *is monotonically decreasing.*

Proof: Suppose that $\gamma_1 > \gamma_2$ *and* $(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N)$ *is an optimal solution of* $F(\gamma_1)$. *Thus,* $F(\gamma_1) = f(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N) - \gamma_1 g(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N) < f(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N) - \gamma_2 g(\overline{\alpha}, \overline{\beta}, \overline{\omega}_1, \ldots, \overline{\omega}_N) \leqslant \max_{\alpha,\beta,\omega_1,\ldots,\omega_N} f(\alpha, \beta, \omega_1, \ldots, \omega_N) - \gamma_2 g(\alpha, \beta, \omega_1, \ldots, \omega_N) = F(\gamma_2)$.

Theorem 5. $F(\gamma) = 0$ *has a unique solution.*

Proof: Based on the above-mentioned theorems, we know $F(\gamma)$ *is continuous as well as decreasing. In addition,* $lim_{\gamma\to+\infty}F(\gamma) = -\infty$ *and* $lim_{\gamma\to-\infty}F(\gamma) = +\infty$.

The procedure of solving this NPPP includes two parts: (1) find such a reasonable parameter γ ($F(\gamma) = 0$), making NPPP equivalent to NFPP; (2) given the parameter γ, solve a polynomial programming problem about the original variables $\alpha, \beta, \omega_1, \ldots, \omega_N$. Our weight adjustment mechanism is an iterative procedure to find the solution of $F(\gamma) = 0$ and the corresponding weights after each iteration of the classification process. We first generate an initial unified classification kernel $\mathbf{K}_j^{(1)}$ with equal weights of $\frac{1}{N}$ to produce an initial classification result on the collaboration multigraph. According to the initial classification result, we then calculate an initial $F(\gamma)$. Since $F(\gamma)$ is a monotonic decreasing function and $F(0) = \max_{\alpha,\beta,\omega_1,\ldots,\omega_N} f(\alpha, \beta, \omega_1, \ldots, \omega_N)$ is obviously non-negative, we start with an initial $\gamma = 0$ and solve the subproblem $F(0)$ by using existing fast polynomial programming model to update the weights $\alpha, \beta, \omega_1, \ldots, \omega_N$. The parameter γ is gradually increased by $\gamma = f(\alpha, \beta, \omega_1, \ldots, \omega_N)/g(\alpha, \beta, \omega_1, \ldots, \omega_N)$ to help the algorithm enter the next round. The algorithm repeats the above-mentioned iterative procedure until $F(\gamma)$ converges to 0.

By assembling different pieces together, we provide the pseudo code of our **AEClass** classifier in Algorithm 1.

Algorithm 1 Activity-Edge Centric Multi-label Classification

Input: a collaboration graph CG, N activity graphs AG_i, a class number K, initial weights $\alpha^{(2)}=\beta^{(2)}=\frac{1}{2}$, $\omega_1^{(1)}=\cdots=\omega_N^{(1)}=\frac{1}{N}$ and a parameter $\gamma^{(1)}=0$.
Output: the predicted label Y_u for the testing instances V_l.

1: Invoke NetClus to partition each of N kinds of activities into K clusters simultaneously;
2: Calculate the category similarity on each AG_i in Eq.(11);
3: Execute the edge classification on CG based on each AG_i in Eqs.(1)-(2);
4: Construct the collaboration multigraph MG;
5: Compute the unified weight $\mathbf{W}_j^{(1)}$ of MG for each c_j in Eq.(3);
6: Calculate the transition probability $\mathbf{T}_j^{(1)}$ for each c_j in Eqs.(4)-(5);
7: Generate the classification kernel $\mathbf{K}_j^{(1)}$ for each c_j in Eq.(6);
8: **for** t=1 **to** $F(\gamma^{(t)})$ converges to 0
9: Calculate the class-membership matrices $\mathbf{X}^{(t)}$ in Eqs.(8)-(10) and $\mathbf{Y}^{(t)}$ in Eqs.(12)-(13);
10: Compute the Macro-F1 score in Eq.(22);
11: Solve $F(\gamma^{(t)})$ in Eq.(27);
12: Update $\omega_1^{(t+1)}$, ..., $\omega_N^{(t+1)}$ if t=1, or update $\alpha^{(t+1)}$, $\beta^{(t+1)}$, $\omega_1^{(t+1)}$, ..., $\omega_N^{(t+1)}$ if t>1;
13: Refine $\gamma^{(t+1)}=f(\alpha^{(t+1)},\beta^{(t+1)},\omega_1^{(t+1)},...,\omega_N^{(t+1)})/g(\alpha^{(t+1)},\beta^{(t+1)},\omega_1^{(t+1)},...,\omega_N^{(t+1)})$;
14: Update $\mathbf{W}_j^{(t+1)}$ in Eq.(3);
15: Adjust $\mathbf{T}_j^{(t+1)}$ in Eqs.(4)-(5);
16: Update $\mathbf{K}_j^{(t+1)}$ in Eq.(17);
17: Return $\mathbf{Y}^{(t)}$ and Y_u.

4. EXPERIMENTAL EVALUATION

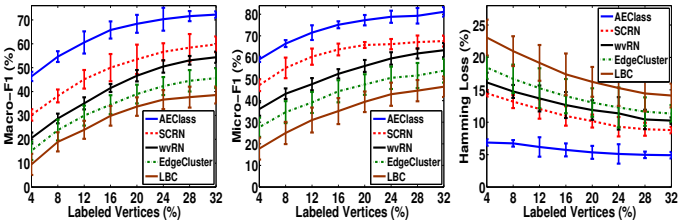
We have performed extensive experiments to evaluate the performance of our **AEClass** classifier on real graph datasets.

4.1 Experimental Datasets

The first real-world dataset is extracted from the DBLP Bibliography data [1]. We build a coauthor graph with highly prolific 100,000 authors from all research areas and 712,834 associated links where vertices represent authors and edges represent their coauthor relationships, and two associated activity graphs: conference graph and term graph. According to [31], we categorize research areas into 24 fields: AI, AIGO, ARC, BIO, CV, DB, DIST, DM, EDU, GRP, HCI, IR, ML, MUL, NLP, NW, OS, PL, RT, SC, SE, SEC, SIM, WWW. We utilize a multi-typed soft clustering framework, NetClus [25], to cluster conferences and terms into 24 categories simultaneously. According to conference's or term's clustering distribution over 24 categories and ranking score in each category, we calculate the similarities between conferences or terms. The classification goal is infer research areas of each author.

Last.fm [2] is a music-oriented online social network. We use the API call *user.getfriends* to collect the list of friends and construct a friendship graph with 50,000 users and 496,611 associated links where vertices represent users and edges denote their friendships. The two activity networks: artist graph and track graph are generated by invoking the API calls *artist.getSimilar* and *track.getSimilar* respectively. By calling the API calls *user.getTopArtists* and *user.getTopTracks*, we classify each friendship edge in terms of the same artists or the same tracks shared by two users. The classification task is to assign each user to a subset of 21 music genres in the database: acoustic, ambient, blues, classical, country, electronic, emo, folk, hardcore, hip hop, indie, jazz, latin, metal, pop, pop punk, punk, reggae, rnb, rock, soul.

[1] http://dblp.uni-trier.de/xml/
[2] http://www.last.fm/api



(a) Macro-F1 (b) Micro-F1 (c) Hamming Loss

Figure 7: Classification Quality on DBLP

The third real dataset is extracted from the Internet Movie Database (IMDb) [3]. We construct a collaboration graph with 10,000 highly prolific actors and 270,227 links where vertices represent actors and edges specify their costar relationships in terms of co-appearance of actors in the same movies. We build a movie activity graph where edges denote co-direct relationship between movies, i.e., movies are directed by the same directors. The objective is to associate each actor with a subset of 22 movie genres: Action, Adventure, Animation, Biography, Comedy, Crime, Documentary, Drama, Family, Fantasy, Film-Noir, History, Horror, Music, Musical, Mystery, Romance, Sci-Fi, Sport, Thriller, War, Western.

4.2 Comparison Methods and Evaluation

We compare **AEClass** with two representative link-based classification algorithms, **LBC** [1], **wvRN** [2], and two recently developed multi-label classifiers, **EdgeCluster** [18], **SCRN** [20]. All four methods perform multi-label classification on a single weighted graph based on the assumption of vertex homophily. The detailed introductions for four methods are presented in Section 5. Note that LBC is originally a multi-class classifier. In order to compare all algorithms, we modify the last step in LBC and use the posterior probability distribution over K classes as the multi-label classification result. AEClass integrates multiple information networks into a unified multigraph with combining both the vertex-centric multi-label classification and the edge-centric multi-label classification based on vertex-edge homophily. It also integrates both the structure affinity and the label vicinity into a unified classifier through dynamic weight tuning mechanism.

Evaluation Measures We use three measures to evaluate the quality of classification results generated by different methods. The first measure is Macro-F1 defined in Definition 3. Given the same definitions in Eq.(18), other two metrics are defined as follows.

$$Micro\text{–}F1 = \frac{2\sum_{j=1}^{K}\sum_{i=l+1}^{N_{CG}} \mathcal{I}(\hat{y}_i^j = 1) y_i^j}{\sum_{j=1}^{K}\sum_{i=l+1}^{N_{CG}} \mathcal{I}(\hat{y}_i^j = 1) + \sum_{j=1}^{K}\sum_{i=l+1}^{N_{CG}} y_i^j} \tag{28}$$

Micro-F1 [32] represents the harmonic mean of micro average of precision and recall. The larger the value, the better the quality.

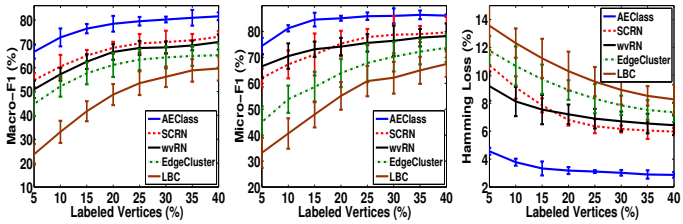
$$HammingLoss = \frac{1}{N_{CG}-l}\sum_{i=l+1}^{N_{CG}} \frac{1}{K}\|\mathcal{I}(\hat{y}_i = 1) \oplus y_i\|_1 \tag{29}$$

where $\oplus$ represents the XOR operation, and $\|\cdot\|_1$ specifies the $l1$-norm. Hamming Loss [33] measures the loss between true labels and predicted labels. The smaller the value, the better the quality.

4.3 Classification Quality

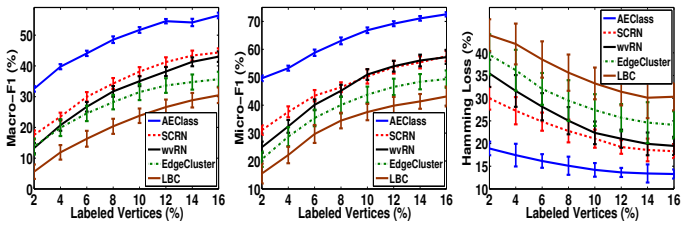
Figures 7-9 exhibit the classification quality on DBLP, Last.fm and IMDb by varying the proportion of labeled vertices respectively. For each proportion of labeled vertices, we average the performance scores over 10 cross-validation folds. The average performance scores with standard deviations of five multi-label classification methods are reported with respect to three evaluation measures of Macro-F1, Micro-F1 and Hamming Loss. We make the following observations on the performances by different methods.

[3] http://www.imdb.com/interfaces



(a) Macro-F1 (b) Micro-F1 (c) Hamming Loss

Figure 8: Classification Quality on Last.fm



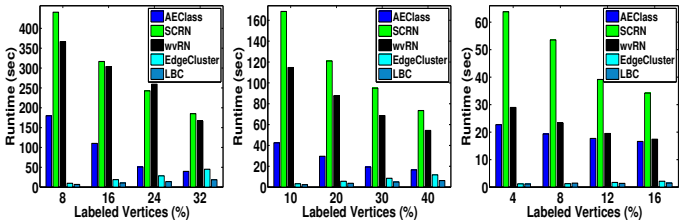
(a) Macro-F1 (b) Micro-F1 (c) Hamming Loss

Figure 9: Classification Quality on IMDb

First, AEClass, SCRN and wvRN significantly outperform LBC and EdgeCluster on all three evaluation measures. We first categorize five multi-label classification methods into non-transductive learning methods and transductive learning methods, based on how they utilize topological structure information. As non-transductive learning methods, both LBC and EdgeCluster only utilize the direct links between vertices in the graph, i.e., one-hop structure information, to produce vertex's features. As transductive learning approaches, AEClass, SCRN and wvRN make full use of both direct links and indirect edges (the circle of friends) between vertices through iterative graph propagation, i.e., multiple-hop structure information, to further improve the classification quality. These results demonstrate the importance of exploiting both direct links and indirect edges for multi-label classification in networked data.

Second, SCRN always outperforms wvRN on three graph datasets. Although SCRN and wvRN exploit the very similar relational inference framework, SCRN improves wvRN by integrating both the network topology and the social context features extracted by EdgeCluster into the classifier. A careful examination reveals that these two approaches are very close in terms of prediction performance in many situations, in spite of the optimization adopted by SCRN. A reasonable explanation is that both of them are only based on the assumption of vertex homophily, i.e., the principle that similar vertices in nature are connected to each other with social links.

Finally, among all five classification methods, AEClass achieves the best classification performance on all three real datasets for all three evaluation measures. Compared to other algorithms, AEClass averagely achieves 14.6% Macro-F1 increase, 12.1% Micro-F1 boost and 5.2% loss reduction on DBLP, 10.2% Macro-F1 growth, 9.9% Micro-F1 increase and 4.1% loss decrease on Last.fm, and 16.7% Macro-F1 increase, 16.2% Micro-F1 boost and 7.5% loss reduction on IMDb, respectively. Note that even if the proportion of labeled vertices is very small, such as 2% and 4%, AEClass still can achieve comparable accuracy on all datasets. Concretely, there are four critical reasons for high accuracy of AEClass: (1) the structure information from associated activity networks boosts the effectiveness of classification. Activity network partition provides us with additional activity labels; (2) the multigraph organization integrates both the vertex-centric multi-label classification based on vertex homophily and the edge-centric multi-label classification based on vertex-edge homophily to leverage the classification performance; (3) Activity network partition captures the inter-dependencies among multiple class labels; and (4) the iterative learning algorithm help the classifier achieve a good balance among different activity-based edge classification schemes and an effective integration of the structure affinity and the label vicinity.



(a) DBLP (b) Last.fm (c) IMDb

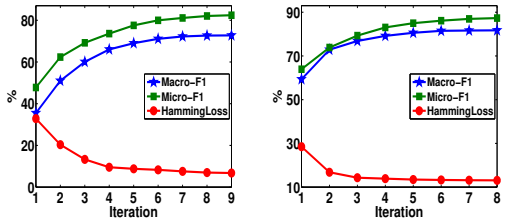
Figure 10: Classification Efficiency

4.4 Classification Efficiency

Figures 10 (a), (b) and (c) present the classification time on DBLP, Last.fm and IMDb with different proportions of labeled vertices respectively. First, LBC has lowest runtime in seconds compared to all other algorithms in all experiments, since it is a logistic regression classifier by aggregating the labels of neighbors as vertex's feature vector. Second, EdgeCluster, a linear SVM classifier with an edge clustering scheme to extract sparse social dimensions, is slightly slower than LBC since the linear SVM approaches generally fall behind the LR methods in speed. Both LBC and EdgeCluster are faster than other three methods because both only utilize the direct links between vertices, i.e., one-hop structure information. In comparison, AEClass, SCRN and wvRN use both direct links and indirect edges (the circle of friends) between vertices, i.e., multiple-hop structure information. Thus, the last three classifiers have higher time complexity than the first two models but they achieve better classification quality. Third, wvRN is consistently faster than SCRN on all three datasets. SCRN improves wvRN by integrating the social dimensions extracted by EdgeCluster into the classifier. This improvement results in an additional computational cost for calculating the class propagation probability of each vertex on each class. Finally, AEClass significantly outperforms the other two transductive learning based multi-label classifiers: SCRN and wvRN. Although SCRN and wvRN execute the classification on a general graph, AEClass does classification on an activity-based collaboration multigraph by augmenting its edges with class labels from each activity graph. There are three main reasons for high efficiency of AEClass: (1) the multigraph organization increases the size of dataset but reduces the computational cost of classification. As we discussed in Subsection 3.2, based on the vertex homophily, no matter which class the current objective is, both SCRN and wvRN need to check each neighbor of a vertex and summarize the labels of all neighbors. In contrast, AEClass only picks up those vertices and links with the same label as the current objective class to execute the inference. Most importantly, AEClass stops the label propagation through irrelevant neighbors (without a link with the same label as the current objective class) in the future iterations; (2) the label vicinity between vertices based on the class-membership distribution over K classes is integrated into the classifier; and (3) we transform the original nonlinear fractional programming problem of multiple weights into a nonlinear parametric programming problem of single variable. According to Theorems 2-5, solving $F(\gamma)$ for a given γ is a polynomial programming problem which can be sped up by existing fast polynomial programming model.

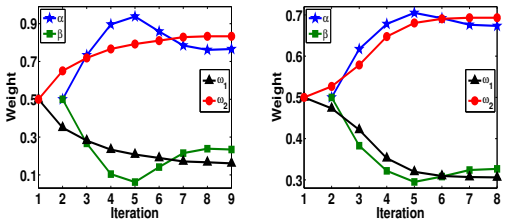
4.5 Classification Convergence

Figure 11 (a) and (b) exhibit the trend of classification convergence in terms of Macro-F1, Micro-F1, and Hamming Loss on DBLP with 4% label nodes and Last.fm with 5% label vertices. Both the Macro-F1 values and the Micro-F1 scores in two figures keep increasing and have concave curves when we iteratively perform the tasks of vertex labeling, weight update and kernel adjustment



(a) DBLP (b) Last.fm

Figure 11: Classification Convergence



(a) DBLP (b) Last.fm

Figure 12: Weight Update

during the classification process. On the other hand, the Hamming Loss values decrease with the classification iterations and have a convex curve. The classification process converges very quickly, usually in eight iterations for Last.fm and nine iterations for DBLP.

Figure 12 (a) and (b) show the tendency of weight update on DBLP and Last.fm respectively. α and β in two figures represent the weights of structure affinity and label vicinity in the unified classifier $\mathbf{K}_j^{(t+1)}$ in Eq.(17) respectively. ω_1 and ω_2 in Figure 12 (a) denote the weights of the conference graph and the term graph respectively. ω_1 and ω_2 in Figure 12 (b) represent the weights of the artist graph and the track graph respectively. We keep the constraints $\alpha+\beta=1$ and $\omega_1+\omega_2=1$ during the classification process. We observe that all the weights converge as the clustering process converges. An interesting phenomenon is that α first increases and then decreases with the iterations and the β curve has a converse trend. A reasonable explanation is that there is lack of enough labeling information at the beginning of classification such that the unified classifier has to rely mostly on the structure affinity to achieve a good classification performance. After a few iterations, we have enough labeling information to utilize both the structure affinity and the label vicinity to classify vertices. An interesting finding is that the term weight is increasing but the conference weight is decreasing with more iterations. A reasonable explanation is that people who have many publications on the same conferences may have different research topics but people who have many papers with the same terms usually have the same research topics. For example, both database papers and data mining papers are published on *VLDB*. Similarly, the track weight increases but the artist weight decreases with more iterations. This is because users who favor the same artists may belong to different music genres since the artists are often related to multiple genres but users who like the same tracks usually belong to the same music genres.

4.6 Case Study

We examine some details of the experiment results on DBLP 100,000 Authors when the proportion of labeled vertices is equal to 32% based on the coauthor graph, the conference graph and the term graph. Table 1 shows the set of authors and their class-membership probabilities after seven iterations based on 24 conference categories and 24 term categories. We only present most prolific DBLP experts in the area of database (DB), data mining (DM), machine learning (ML) and information retrieval (IR). The class-membership scores in Table 1 are normalized by different (conference or term) categories for each author. We observe that the predicted class memberships of authors are consistent with their

Author/Class	DB	DM	ML	IR
Peter L. Bartlett	0.019	0.039	0.938	0.004
Elisa Bertino	0.738	0.054	0.028	0.180
Andrei Z. Broder	0.037	0.097	0.015	0.851
Michael J. Carey	0.965	0.029	0.006	0.023
W. Bruce Croft	0.054	0.007	0.037	0.902
David J. DeWitt	0.912	0.057	0.004	0.027
Inderjit S. Dhillon	0.030	0.409	0.457	0.104
Christos Faloutsos	0.321	0.500	0.031	0.147
Jiawei Han	0.391	0.463	0.045	0.100
H. V. Jagadish	0.850	0.056	0.009	0.048
Michael I. Jordan	0.007	0.062	0.917	0.014
Daphne Koller	0.026	0.045	0.915	0.013
Vipin Kumar	0.120	0.622	0.199	0.059
Bing Liu	0.086	0.427	0.266	0.220
Hector Garcia-Molina	0.788	0.010	0.016	0.186
C. J. van Rijsbergen	0.003	0.051	0.024	0.922
Michael Stonebraker	0.946	0.013	0.007	0.034
Jeffrey D. Ullman	0.824	0.065	0.064	0.047
Philip S. Yu	0.342	0.496	0.044	0.118
Mohammed J. Zaki	0.148	0.672	0.057	0.123

Table 1: Class-membership Probabilities of Authors Based on Conference and Keyword Partitions from DBLP

actual research areas. For those experts with unique research areas, such as *Michael J. Carey* and *Michael Stonebraker*, the primary research areas for them in the predicted result are obviously consistent with their actual research areas; For those researchers known to work in multiple research areas, the predicted class-membership distributions also correspond to their current research activities. For example, both *Jiawei Han* and *Philip S. Yu* are experts on data mining and database, though their *DM* probabilities are slightly higher since each of them and their circle of co-authors have more *DM* papers. This table also shows that each author has a class-membership score in each category. This demonstrates that our AEClass model can make each author quickly reach each class label.

5. RELATED WORK

Node classification in networked data has attracted active research in the last decade [1–9]. LBC [1] is a network-only derivative of the link-based classifier which creates a feature vector for a node by aggregating the labels of neighboring nodes, and then uses logistic regression to build a discriminative model based on these feature vectors. wvRN [2] presented a weighted-vote relational neighbor classifier to solve link-based classification problems based solely on the class labels of linked neighbors. DYCOS [6] exhibited a node classification model in dynamic information networks with both text content and links. RankClass [7] integrates classification and ranking in a mutually enhancing process to provide class summaries for heterogeneous information networks.

Multi-label classification is gaining attention in recent years [10–16]. Read et al. [12] reduces the complexity and potential for error with a pruning procedure to focus on core relationships within multi-label sets. IBLR [13] proposed a multi-label classification approach to combine model-based and similarity-based inference with the estimation of optimal regression coefficients. LEAD [15] decomposes a multi-label learning task into a set of single-label classification problems with a Bayesian network to encode the conditional dependencies of labels as well as the feature set. Guo and Gu [16] proposed a generalized conditional dependency network for model training using binary classifiers and label predictions using Gibbs sampling inference.

Multi-label classification in networked data has been extensively studied in recent years [17–22]. Sun et al. [17] presented a hypergraph spectral learning formulation for multi-label classification, where a hypergraph is constructed to exploit the correlation information among different labels. EdgeCluster [18] presented a social-dimension based approach for collective behavior prediction with

an edge clustering scheme to extract sparse social dimensions and a linear SVM classifier for discriminative learning. SCRN [20] is a multi-label iterative relational neighbor classifier by considering both network topology and social context features. PIPL [21] facilitates the multi-label learning process by mining label correlations and instance correlations from the heterogeneous networks.

Recent works on heterogeneous social network analysis [7,9,23–30] combine links and content into heterogeneous information networks to improve the quality of querying, ranking and clustering. Cai et al. [23] proposed to learn an optimal linear combination of different relations on heterogeneous social networks in terms of their importance on a certain query. GenClus [28] proposed a model-based method for clustering heterogeneous networks with different link types and different attribute types. Yu et al. [29] presented a query-driven discovery system for finding semantically similar substructures in heterogeneous networks.

To our knowledge, this work is the first one to address the problem of activity-edge centric multi-label classification of heterogeneous multigraph with the prior knowledge of multiple activity graphs by dynamically adjusting their individual contributions.

6. CONCLUSIONS

We have presented an edge-centric multi-label classification approach for mining heterogeneous information networks. First, we integrate the primary social network and multiple associated activity networks into a unified multigraph with edge classification. Second, we combine both the structure affinity and the label vicinity based on multiple activity networks into a unified classifier. Third, an iterative learning algorithm is proposed dynamically refine the classification result by continuously adjusting the weights on different activity-based edge classification schemes from multiple activity graphs, while constantly learning the contributions of the structure affinity and the label vicinity in the unified classifier.

Acknowledgement. This material is based upon work partially supported by the NSF under Grants IIS-0905493, CNS-1115375, IIP-1230740, and a grant from Intel ISTC on Cloud Computing.

7. REFERENCES

[1] Q. Lu and L. Getoor. Link-based classification. In *ICML'03*.

[2] S. A. Macskassy and F. Provost. A simple relational classifier. In *MRDM*, pages 64–76, 2003.

[3] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Scholkopf. Learning with local and global consistency. In *NIPS*, 2003.

[4] J. Neville and D. Jensen. Relational dependency networks. *Journal of Machine Learning Research*, 8:653–692, 2007.

[5] S. A. Macskassy and F. Provost. Classification in networked data: A toolkit and a univariate case study. *Journal of Machine Learning Research*, 8:935–983, 2007.

[6] C. C. Aggarwal and N. Li. On Node Classification in Dynamic Content-based Networks. In *SDM*, 2011.

[7] M. Ji, J. Han, and M. Danilevsky. Ranking-based classification of heterogeneous information networks. In *KDD*, pages 1298–1306, 2011.

[8] S. Bhagat, G. Cormode, and S. Muthukrishnan. Node classification in social networks. *Social Network Data Analytics*, pages 115–148, 2011.

[9] X. Kong, P. S. Yu, Y. Ding, and D. J. Wild. Meta path-based collective classification in heterogeneous information networks. In *CIKM*, pages 1567–1571, 2012.

[10] S. Godbole and S. Sarawagi. Discriminative methods for multi-labeled classification. In *PAKDD*, 2004.

[11] G. Chen, Y. Song, F. Wang, and C. Zhang. Semi-supervised multi-label learning by solving a sylvester equation. In *SDM*, Atlanta, GA, April 24-26 2008.

[12] J. Read, B. Pfahringer, and G. Holmes. Multi-label classification using ensembles of pruned sets. In *ICDM'08*.

[13] W. Cheng and E. Hullermeier. Combining instance-based learning and logistic regression for multilabel classification. *Machine Learning*, 76(2-3):211–225, 2009.

[14] K. Dembczynski, W. Cheng, and E. Hullermeier. Bayes optimal multilabel classification via probabilistic classifier chains. In *ICML*, Haifa, Israel, June 21-24 2010.

[15] M.-L. Zhang and K. Zhang. Multi-label learning by exploiting label dependency. In *KDD*, 999–1008, 2010.

[16] Y. Guo and S. Gu. Multi-label classification using conditional dependency networks. In *IJCAI*, 2011.

[17] L. Sun, S. Ji, and J. Ye. Hypergraph spectral learning for multi-label classification. In *KDD*, 668–676, 2008.

[18] L. Tang and H. Liu. Scalable learning of collective behavior based on sparse social dimensions. In *CIKM*, 2009.

[19] S. Peters, Y. Jacob, L. Denoyer, and P. Gallinari. Iterative multi-label multi-relational classification algorithm for complex social networks. *Social Network Analysis and Mining*, 2(1):17–29, 2012.

[20] X. Wang and G. Sukthankar. Multi-label relational neighbor classification using social context features. In *KDD*, 2013.

[21] X. Kong, B. Cao, and P. S. Yu. Multi-label classification by mining label and instance correlations from heterogeneous information networks. In *KDD*, pages 614–622, 2013.

[22] B. Wang, Z. Tu, and J. K. Tsotsos. Dynamic label propagation for semi-supervised multi-class multi-label classification. In *ICCV*, 2013.

[23] D. Cai, Z. Shao, X. He, X. Yan, and J. Han. Community mining from multi-relational networks. In *PKDD*, 2005.

[24] Y. Zhou, H. Cheng, and J. X. Yu. Graph clustering based on structural/attribute similarities. In *VLDB*, 718–729, 2009.

[25] Y. Sun, Y. Yu, and J. Han. Ranking-based clustering of heterogeneous information networks with star network schema. In *KDD*, pages 797–806, 2009.

[26] T. Yang, R. Jin, Y. Chi, and S. Zhu. Combining link and content for community detection: a discriminative approach. In *KDD*, pages 927–936, 2009.

[27] Y. Zhou, H. Cheng, and J. X. Yu. Clustering large attributed graphs: An efficient incremental approach. In *ICDM*, 2010.

[28] Y. Sun, C. C. Aggarwal, and J. Han. Relation strength-aware clustering of heterogeneous information networks with incomplete attributes. *PVLDB*, 5(5):394–405, 2012.

[29] X. Yu, Y. Sun, P. Zhao, and J. Han. Query-driven discovery of semantically similar substructures in heterogeneous networks. In *KDD*, 1500–1503, 2012.

[30] Y. Zhou and L. Liu. Social influence based clustering of heterogeneous information networks. In *KDD*, 2013.

[31] T. Chakraborty, S. Sikdar, V. Tammana, N. Ganguly, and A. Mukherjee. Computer science fields as ground-truth communities: Their impact, rise and fall. In *ASONAM*, 2013.

[32] R.-E. Fan and C.-J. Lin. A study on threshold selection for multi-label classification. In *Tech Report*, National Taiwan University, 2007.

[33] X. Zhang, Q. Yuan, S. Zhao, W. Fan, W. Zheng, and Z. Wang. Multi-label classification without the multi-label cost. In *SDM*, 2010.