Multi Task Learning on Multiple Related Networks

MC Prakash
Dept. of Computer Science
Michigan State University
East Lansing
Michigan
mandayam@msu.edu

Pang-Ning Tan
Dept. of Computer Science
Michigan State University
East Lansing
Michigan
ptan@msu.edu

Anil K. Jain
Dept. of Computer Science
Michigan State University
East Lansing
Michigan
jain@cse.msu.edu

ABSTRACT

With the rapid proliferation of online social networks, the need for newer class of learning algorithm to simultaneously deal with multiple related networks has become increasingly important. This paper proposes an approach for multi-task learning in multiple related networks, where in we perform different tasks such as classification on one network and clustering on the other. We show that the framework can be extended to incorporate prior information about the correspondences between the clusters and classes in different networks. We have performed experiments on real-world data sets to demonstrate the effectiveness of the proposed framework.

Categories and Subject Descriptors

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

General Terms

Algorithms

Keywords

Clustering, Classification, Multi task learning, Link Mining

1. INTRODUCTION

With the rapid proliferation of diverse online social networks, the need to integrate and analyze relational data from multiple networks has become increasingly important. Typical learning problems for relational data include community detection (or clustering), collective classification, link prediction, influence maximization, and anomaly detection. However, the focus of previous research has been on learning a single task on a single network or on a k-partite graph constructed from a multi-network with heterogeneous nodes.

This paper departs from previous research by focusing on learning multiple tasks (clustering and classification) in different networks. For example, one might be interested in classifying the postings made to an online message forum while simultaneously clustering the network of users who posted the messages or comments. Multi-task learning can also be performed on networks from different domains. For example, suppose we have partially classified the users on Facebook. Can we use this information to help cluster the users on MySpace or other social networking Web sites?

One approach to solve this problem would be to combine all the graphs into one giant network and apply existing state of the art clustering or classification algorithms to the whole network. However such an approach has many limitations. First, the number of classes in one network may be different than the number of communities (clusters) in another network. By applying a single algorithm to the entire network, we have no control over the number of clusters or classes that will be found in each induced subgraph of the network. Furthermore, aggregating the graphs into one large network will lose information about the inherent variabilities between different graphs.

Another approach would be to perform the learning task independently on each network. An obvious limitation of this approach is that it does not fully utilize the link information from other related networks. More importantly, if the learning tasks are somewhat related and there is prior knowledge about the relationships between the clusters and classes in different networks, this approach will not be able to share and utilize this information. This motivates the need to develop a joint algorithm for multi-task learning in multiple related networks.

There have been recent efforts for clustering multiple graphs. Zhou et al. [6] proposed a new method to combine multiple graphs to measure document similarities, where different factorization strategies are used based on nature of different graphs. Tang et al. [4] proposed a linked matrix factorization approach for fusing information from multiple graph sources. Chen et al. [1] presented a co-classification framework for Web spam and spammer detection in social media based on the maximum margin principle. They demonstrated that joint classification strategy is more effective than independent detection strategy. However, to best of our knowledge none of the previous studies has worked on learning multiple tasks such as classification and clustering simultaneously on multiple related networks.

2. PRELIMINARIES

Let $\mathcal{G} = (\bigcup_{i=1}^{M} V_i, \mathcal{E}_s \cup \mathcal{E}_d)$ be the graph representation of a multi-network, where each V_i corresponds to a set of nodes of a specific type and M is the number of node types.

CIKM'10, October 26–30, 2010, Toronto, Ontario, Canada.
Copyright 2010 ACM 978-1-4503-0099-5/10/10 ...$10.00.

Furthermore, $\mathcal{E}_s$ is the set of links connecting the same type of nodes whereas $\mathcal{E}_d$ is the set of links connecting nodes of different types. The multi-network forms a k-partite graph if $\mathcal{E}_s = \emptyset$. Each set of nodes of the same type also induces a subgraph $\mathcal{G}_i = (V_i, E_i)$, where $E_i \subseteq V_i \times V_i$ and $\mathcal{E}_s = \bigcup_{i=1}^{T} E_i$. We may consider the induced subgraph $\mathcal{G}_i$ as a network of homogeneous nodes within the multi-network.

Definition 1 (Multi-task Multi-network Learning). *Given $\mathcal{G} = (\bigcup_{i=1}^{M} V_i, \mathcal{E}_s \cup \mathcal{E}_d)$, the multi-task multi-network learning problem is to solve M learning tasks, where each task is associated with an induced subgraph $\mathcal{G}_i = (V_i, E_i)$ of $\mathcal{G}$.*

For brevity, the framework presented in this paper focuses on a multi-network with $M = 2$, though it can be generalized to graphs containing more than two types of nodes. To simplify the notation, let $\mathcal{G}_1 = (V_1, E_1)$ and $\mathcal{G}_2 = (V_2, E_2)$ be the induced subgraphs of $\mathcal{G}$ containing nodes from V_1 and V_2, respectively. Also, let $\mathcal{G}_{12} = (V_1 \cup V_2, \mathcal{E}_d)$ denote a bipartite graph with links connecting between nodes in V_1 to those in V_2. The number of nodes in each network are denoted as $|V_1| = n$ and $|V_2| = m$, respectively.

This paper focuses on a multi-task learning problem in which the tasks involve classification and clustering of nodes in two related networks. Without loss of generality, we assume that the clustering task is performed on network $\mathcal{G}_1$ and the classification task on network $\mathcal{G}_2$. Let k_1 be the number of communities (clusters) in $\mathcal{G}_1$ and k_2 be the number of classes in $\mathcal{G}_2$. Let A, B, and C be the adjacency matrices associated with the graphs $\mathcal{G}_1$, $\mathcal{G}_2$ and $\mathcal{G}_{12}$, respectively. We assume that the first l nodes in $\mathcal{G}_2$ are labeled while the remaining $m - l$ nodes are unlabeled. The true class information is encoded in an $l \times k_2$ binary matrix L, such that $L_{ij} = 1$ if the node $v_{2i} \in V_2$ belongs to class j and zero otherwise.

Our approach is to generate a pseudo label matrix $X \in \mathcal{R}^{n \times k_1}$ for the nodes in $\mathcal{G}_1$ such that the i^{th} row in X gives the cluster membership of node v_{1i} in each of the k_1 clusters. Similarly, a pseudo label matrix $Y \in \mathcal{R}^{m \times k_2}$ is generated for the nodes in $\mathcal{G}_2$ such that the i^{th} row of Y gives its membership in each of the k_2 labels.

3. THE PROPOSED FRAMEWORK

This section outlines our proposed framework for simultaneous clustering and classification of multiple related networks. Specifically, we employ a matrix factorization approach [2] to solve both tasks by optimizing the following joint objective function:

$$\begin{aligned}\min_{X,U,V,Y,W} \quad & D(A \parallel XUX^T) + D(C \parallel XVY^T) \\ + \quad & D(B \parallel YWY^T) + \beta D(L \parallel Y_l) \end{aligned} \quad (1)$$

where the superscript T denote the matrix transpose operation and $D(P \| Q) = \sum_{ij} P_{ij} \log\left(\frac{P_{ij}}{Q_{ij}}\right) - P_{ij} + Q_{ij}$ is the Kullback-Leibler distance between P and Q. The first term in the objective function deals with the clustering of nodes in $\mathcal{G}_1$ by factorizing the adjacency matrix A into a product involving the pseudo label matrix X. The last two terms deal with the classification of nodes in $\mathcal{G}_2$ by estimating the pseudo label matrix Y, taking into account both the link structure (B) and class information (L). It should be noted that the last term in the objective function does not apply to unlabeled nodes in network $\mathcal{G}_2$. Thus, Y_l is an $l \times k_2$ sub-matrix of Y. Finally, the second term in the objective function is used to learn the relationship between the clusters found in network $\mathcal{G}_1$ and the classes obtained for network $\mathcal{G}_2$.

Clearly, if the pseudo label matrix Y is known, we can augment this information to estimate the pseudo label matrix X for network $\mathcal{G}_1$ by minimizing the following objective function

$$\mathcal{L}_1 : \min_{X,U,V} D(A \parallel XUX^T) + D(C \parallel XVY^T) \quad (2)$$

Similarly, if the community membership information in network $\mathcal{G}_1$ is available, then it can be used as additional information for classifying the nodes in $\mathcal{G}_2$. This is accomplished by minimizing the following objective function

$$\mathcal{L}_2 : \min_{Y,W,V} D(B \parallel YWY^T) + D(C \parallel XVY^T) + \beta D(L \parallel Y_l) \quad (3)$$

Thus, we may use an alternating minimization scheme to solve the optimization problem. Since we have five parameters to estimate (X, Y, U, W, and V), we iteratively update the value of each parameter by fixing the values of the remaining four parameters. The update formula for each parameter is computed using a gradient descent approach. We omit the details due to space limitations and present the multiplicative update formula below:

$$X_{ij} = X_{ij} \frac{\sum_a \left(\frac{A_{ia}[XU^T]_{aj}}{[XUX^T]_{ia}} + \frac{A_{ai}[XU]_{aj}}{[XUX^T]_{ai}}\right) + \sum_a \frac{C_{ia}[YV^T]_{aj}}{[XVY^T]_{ia}}}{\left(\sum_a [XU + XU^T + \sum_a [YV^T]_{aj}\right)} \quad (4)$$

The update formula for Y_{ij} depends on whether the node is labeled or not. For $i > l$ (unlabeled nodes)

$$Y_{ij} = Y_{ij} \frac{\sum_a \left(\frac{B_{ia}[YW^T]_{aj}}{[YWY^T]_{ia}} + \frac{B_{ai}[YW]_{aj}}{[YWY^T]_{ai}}\right) + \sum_a \frac{C_{ai}[XV]_{aj}}{[XVY^T]_{ai}}}{\left(\sum_a [YW + YW^T + \sum_a [XV]_{aj}\right)} \quad (5)$$

whereas for labeled nodes ($i \leq l$), we need to add $\beta \sum_{a=1}^{k_2} \frac{L_{ia}}{Y_{ia}}$ and $\sum_a [I_{k_2}]_{aj}$ respectively to the numerator and denominator of update formula (5), where I_k is identity matrix Similarly, the update formula for matrices U, W, and V are computed as follows:

$$U_{ij} = U_{ij} \left[\frac{\sum_{a=1}^{M} \sum_{b=1}^{N} \frac{A_{ab}}{[XUX^T]_{ab}} X_{ai} X_{bj}}{\sum_{a=1}^{M} \sum_{b=1}^{N} X_{ai} X_{bj}} \right] \quad (6)$$

$$W_{ij} = W_{ij} \left[\frac{\sum_{a=1}^{M} \sum_{b=1}^{N} \frac{B_{ab}}{[YWY^T]_{ab}} Y_{ai} Y_{bj}}{\sum_{a=1}^{M} \sum_{b=1}^{N} Y_{ai} Y_{bj}} \right] \quad (7)$$

$$V_{ij} = V_{ij} \left[\frac{\sum_{a=1}^{M} \sum_{b=1}^{N} \frac{C_{ab}}{[XVY^T]_{ab}} X_{ai} Y_{bj}}{\sum_{a=1}^{M} \sum_{b=1}^{N} X_{ai} Y_{bj}} \right] \quad (8)$$

We first randomly initialize tthe matrices X, Y, U, W, and V to some non-negative entries and then iteratively update the matrices according to the update formula given above. The process is repeated until a specified maximum number of iterations is reached.

3.1 Incorporating Prior Information

Often times, we may have additional information about the relationship between the classes and clusters in the different networks. In what follows we give a motivation to incorporate this prior information into the objective function.

Example 1. *Consider a multi-network comprising of a citation network between research articles and a co-authorship network between researchers. Suppose we would like to classify the articles according to the following topics:* ***Computer Vision, Pattern Recognition, Artificial Intelligence, Cell Biology,*** *and* ***Genetics****. Similarly, we are interested in classifying the authors according to their research disciplines. Since an author may work on multiple related topics (Computer Vision and AI or Cell Biology and Genetics), therefore the article classes are not reflected as it is in the author network, rather they are further grouped into coarser clusters, namely,* ***Computer Science*** *and* ***Biological Science****. The author cluster (****Computer Science****) is related to the first three article clusters, while* ***Biological Science*** *is related to* ***Cell Biology*** *and* ***Genetics****. We expect such prior information will enhance the joint clustering results. This information can be encoded in a 5×2 prior matrix:*

$$P = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}$$

where the rows are the article categories and the columns are the author clusters.

To incorporate prior, we first need to interpret the role of the V matrix in the objective function. This matrix is estimated from the KL divergence term, $D(C \parallel XVY^T)$, where C is the adjacency matrix representation of the links in the bipartite graph $\mathcal{G}_{12}$. Since X encodes the cluster membership of the nodes in $\mathcal{G}_1$ and Y encodes the class membership of the nodes in $\mathcal{G}_2$, we could interpret the role of matrix V as capturing the relationship between the clusters in $\mathcal{G}_1$ and the classes in $\mathcal{G}_2$.

Let P be the prior information about the relationship between the clusters and classes. To incorporate this information, we modify the objective function (1) as follows:

$$\begin{aligned} &\min_{X,U,V,Y,W} D(A \parallel XUX^T) + D(B \parallel YWY^T) \\ &+ \quad \beta D(L \parallel Y_l) + D(C \parallel X(\lambda V + (1-\lambda)P)Y^T), \quad (9) \end{aligned}$$

where λ is a user-specified parameter that controls the tradeoff between fitting V directly to the data and fitting V to the prior matrix P. If $\lambda = 0$, then the correspondence between clusters is given by the prior matrix. If $\lambda = 1$, then the formulation reduces to the original framework given in (1). In situations where the actual proportion of links between nodes from each cluster in $\mathcal{G}_1$ and each class in $\mathcal{G}_2$ is unknown, we may use a non-informative prior with P_{ij} is 0 or 1 indicating whether the i^{th} cluster is expected to be related to the j^{th} class (see Example 1).

4. EXPERIMENTAL EVALUATION

This section presents the results of applying the proposed framework to the multi-task learning problem on real-world network data. As a baseline, we use the normalized cut (`Ncut`) algorithm by Shi and Malik [3] for clustering and the label propagation algorithm with local and global consistency (`LGC`) by Zhou et al. [5] for classification. For a fair comparison, we applied each baseline algorithm on the entire multi-network $\mathcal{G}$ (instead of $\mathcal{G}_1$ and $\mathcal{G}_2$ separately). In each experiment, we set the proportion of labeled nodes in one of the two network to 20%. We use the normalized mutual information (NMI) measure to evaluate clustering results and accuracy to evaluate classification results. We denote our proposed multi-task, multi-network learning framework as `Joint` or `Joint + Prior` in the remainder of this section.

4.1 Wikipedia Data

Two networks, namely the article networks and editor networks are available on Wikipedia. The Wikipedia articles are chosen from four broad topics—Biology, Natural Science, Computer Science and Social Science. Each of the topics are further subdivided into subtopics, as shown in Table 1.

Table 1: Data Category and Sub Category

Category	Sub-Categories
User clusters	Article clusters
Political Science	Civil-Rights Liberties(878); Imperialism(601); Nationalism(368)
Natural Science	Physics(568); Earth Sciences(513) Astronomy(613)
Computer Science	Algorithms(112); Operating Systems(395); Architecture(350)
Biology	Zoology(392); Anatomy(897) Cell-Biology(716);

We randomly sampled 6403 articles and 5361 of their corresponding editors to form the Wikipedia multi-network. Our task is to classify each article into one of the 12 possible sub-categories and to partition the editors into 4 clusters.

Table 2: Clustering results of Wikipedia editors.

User network	4 clusters (NMI)
Ncut on editor network only	0.08
Ncut to entire multi-network	0.01
Joint without prior	0.30
Joint with Prior	0.37

Table 3: Classification results of Wikipedia articles.

Article network	12 classes (Accuracy)
LGC on article network only	0.87
LGC on entire multi-network	0.85
Joint without prior	0.77
Joint with Prior	0.82

As shown in Table 2, the independent clustering of user network gives very bad results compared to the `Joint` approach. The cluster NMI increases from 0.08 to 0.37. However, this additional gain comes at the expense of reduced classification accuracy on the article network. Applying `LGC` on article network alone gives an accuracy of 0.87 which reduces to 0.85, when applied to the entire multi-network. This is because the class information provided by the article network is more useful than the "coarse-level" cluster information provided by the editor network. Furthermore, a user typically contributes to articles across different categories which makes it difficult to decide his/her actual class label. We currently assigned the user to the category to which he/she has made the most contributions. In fact, it is because of this problem, it is difficult to acquire the label information in user network, and thus, clustering becomes a necessary task.

4.1.1 Wikipedia and Digg Data

Here we present the results of applying the proposed framework to multi-network constructed from different domains. We cluster the users of Digg.com[1] and classify the editors of Wikipedia.org First, articles from Wikipedia are chosen from only three broad topics — Natural Science, Computer Science and Social Science and a sample of 4320 editors are chosen. We then sampled 5670 Digg users who have bookmarked URLs on the following three topics: *Politics, Computer Science*, and *Natural Science*. We formed a Digg user-user link from the user-URL matrix. Two Digg users are linked if they have at least τ URLs in common. Finally, links between Wikipedia editors and Digg users are established using the contents of Wikipedia articles and Digg's URL description. Specifically, the weight of the link corresponds to the cosine similarity between the words in the title and description of a URL and the words that appear in the content of a Wikipedia article. Figure 1 shows the spy plot for the Digg user and Wikipedia editor networks. Careful observation of the spy plots reveals three distinct clusters/classes among both networks. However the Digg information in user network, and thus, clustering becomes a necessary task.

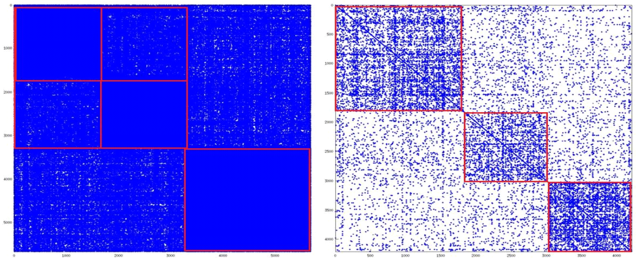

Figure 1: Adjacency matrix plot for Digg user (Left) and Wikipedia editor (Right) networks (best viewed in color).

We first performed `Ncut` on the Digg data alone and `Ncut` on the overall network (Digg + Wikipedia + links between them). The confusion matrix is given in Table 4. As can be seen in the adjacency matrix plot, the first two clusters are noisy and heavily interlinked. So we obtain only two predominant clusters.

Table 4: Confusion matrix for Digg clustering using `Ncut`

`Ncut` on Digg				`Ncut` on Digg + Wikipedia		
1171	3	474		167	1481	0
1149	0	486		997	638	0
392	0	1995		1952	435	0
NMI - 0.143				NMI - 0.185		

We apply the `LGC` algorithm to propagate labels in the Wikipedia data set. The results are summarized in Table 5. The noisy Digg data has degraded the performance of `LGC` on the Wikipedia network. Propagating labels only on Wikipedia data set gives an accuracy of 0.71, which reduces to 0.66 when applied to the multi-network.

Table 5: Confusion matrix for Wikipedia classification using `LGC`

LGC on Wikipedia				LGC on Digg + Wikipedia		
1463	203	150		1688	78	50
350	690	159		710	416	73
232	100	859		450	56	685
Accuracy - 0.71				Accuracy - 0.66		

The presence of noise on the Digg user network combined with noisy links between the Wikipedia and Digg networks resulted in poor performance of the joint learning algorithm. The number of clusters obtained is less than the number we expect. However, by incorporating the prior matrix $P = I_3$, ensures that we obtain three clusters on each network. The results are shown in table below. Clearly, the `Joint + Prior` results are significantly better than both Ncut and LGC.

Table 6: Confusion matrix for clustering Digg users and classifying Wikipedia editors using `Joint` with `prior`

Digg - Cluster				Wiki - Classify		
1246	280	122		1710	84	22
136	1278	221		298	770	131
227	103	2057		55	118	1018
NMI = 0.44				Accuracy - 0.83		

5. CONCLUSION

In this paper we have given a framework to perform multi task learning on multiple related networks. We have also introduced the idea of using a prior to guide the clustering process. We have demonstrated a practical use of our algorithm by identifying similar communities on different network domains namely Digg and Wikipedia.

6. ACKNOWLEDGEMENT

This material is based upon work supported in part by ONR grant number N00014-09-1-0663 and ARO grant number W911NF-09-I-0566.

7. REFERENCES

[1] F. Chen, P. N. Tan, and A. K. Jain. A co-classification framework for detecting web spam and spammers in social media web sites. In *Proc of CIKM*, 2009.

[2] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Proc of NIPS*,2000.

[3] J. Shi and J. Malik. Normalized cuts and image segmentation. In *Proc of CVPR*, 1997.

[4] W. Tang, Z. Lu, and I. Dhillon. Clustering with multiple graphs. In *Proc of ICDM*, 2009.

[5] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency". In *Proc of NIPS*, 2004.

[6] D. Zhou, S. Zhu, K. Yu, X. Song, B. L. Tseng, H. Zha, and C. L. Giles. Learning multiple graphs for document recommendations. In *Proc of WWW '08*.

[1] www.digg.com is a popular social news Web site