Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence

Marginalized Denoising for Link Prediction and Multi-label Learning

Zheng Chen[1,2], Minmin Chen[3], Kilian Q. Weinberger[2], Weixiong Zhang[2,1]
[1]Institute for Systems Biology, Jianghan University, Wuhan, Hubei 430056, China.
[2]Department of Computer Science and Engineering, Washington University in St. Louis, St. Louis, MO 63130, USA.
[3]Criteo Lab, Palo Alto, CA, USA
zheng.chen@wustl.edu, m.chen@criteo.com, {kilian, weixiong.zhang}@wustl.edu

Abstract

Link prediction and multi-label learning on graphs are two important but challenging machine learning problems that have broad applications in diverse fields. Not only are the two problems inherently correlated and often appear concurrently, they are also exacerbated by incomplete data. We develop a novel algorithm to solve these two problems jointly under a unified framework, which helps reduce the impact of graph noise and benefits both tasks individually. We reduce multi-label learning problem into an additional link prediction task and solve both problems with marginalized denoising, which we co-regularize with Laplacian smoothing. This approach combines both learning tasks into a single convex objective function, which we optimize efficiently with iterative closed-form updates. The resulting approach performs significantly better than prior work on several important real-world applications with great consistency.

Introduction

We study two of the most prominent machine learning tasks on graphs: link prediction (LP) and multi-label learning (MLL). On an abstract level the two problems are similar: LP aims to impute missing *links* and MLL aims to impute missing *labels*. In real world applications, the two problems do indeed often appear hand-in-hand. Consider as an example protein-protein interaction (PPI) networks, which represent proteins bound to form protein complexes so as to exert their functions. Here, inferring interactions among proteins is an LP problem (Jansen et al. 2003) and predicting protein functions or annotations is an MLL task (Vazquez et al. 2003). Similarly, in social networks, LP can be used to recommend friends to one another (Liben-Nowell and Kleinberg 2007), and MLL can be used to infer properties about users (*e.g.* age group, profession, and interests) from their social connections.

Data in such applications is notoriously incomplete and usually has *both*, partially observed edges and partially observed labels. In PPI networks, for instance, it is well known that the function annotations and the link structure of even the most well-studied genes in model species are still quite sparse. Similarly, in social networks only a subset of friends explicitly connect one another (missing edges) and only a small fraction of users actually completes the full profile information (missing labels).

Although LP and MLL are inherently interwoven, so far they have been mostly considered to be unrelated problems. In this work we show that both problems can be solved jointly and can substantially benefit from each other. In biology, for example, the (partially) known protein function annotations (labels) have been exploited as a major source of information for predicting PPI (edges) (Jansen et al. 2003). Meanwhile, the functions of a protein are often determined by which other proteins it interact with to form protein complexies, and thus PPI (edges) are commonly used to predict protein functions (labels) (Kourmpetis et al. 2010). Similarly in social networks, adding missing friendship edges will help predict interests of users, which are often shared amongst friends. Likewise, interests of users can be utilized to predict hidden links of friendship.

The most obvious approach to exploit such reciprocal relationship between LP and MLL is to alternately apply LP and MLL algorithms and use the output of one method to facilitate the other (Bilgic, Namata, and Getoor 2007). However, it is non-trivial to determine how frequently and to what extend information should be exchanged between LP and MLL. Most importantly, the alternating iterations between the two tasks are not guaranteed to converge.

We introduce a highly novel approach to combine the MLL and LP problem into a single joint objective with the recently introduced marginalized denoising framework (Chen et al. 2012; Chen, Zheng, and Weinberger 2013; Chen and Zhang 2014). We phrase both problems as instances of graph denoising and co-regularize the predictions with Laplacian graph regularization. Our resulting algorithm, which we refer to as Marginalized Label and Link Prediction (MLLP), is jointly convex and can be optimized very efficiently. Different from prior work (Wang, Huang, and Ding 2011), which essentially uses LP formulation to solve a MLL problem, we incorporate both problems into one single loss function and let them benefit from each other as information naturally flows both ways. We evaluated MLLP on three real-world applications - one social network and two PPI networks. MLLP outperforms every competing prior work consistently across all our benchmark data sets.

Notation

We consider the problems of LP and MLL on a graph $\mathcal{G} = (\mathbf{G}, \mathbf{Y})$ of n nodes in a transductive learning setting. $\mathbf{G} \in \{0,1\}^{n \times n}$ denotes a partially observed relational graph between nodes, where $G_{ij} = 1$ iff we observe a link between nodes i and j, and $G_{ij} = 0$, otherwise. For simplicity, we consider binary graphs throughout the paper, but in practice our method can be easily extended to weighted graphs, where G_{ij} takes real-valued edge weights. $\mathbf{Y} \in \{0,1\}^{K \times n}$ represents partially observed labels of nodes, where K is the total number of labels and $Y_{ki} = 1$ iff node i is observed to be associated with the k^{th} label, and $Y_{ki} = 0$, otherwise. Note that each node can be associated with more than one label. We denote the i^{th} column of $\mathbf{Y}$, *i.e.* the label vector of node i, as $\mathbf{y}_i$ and without loss of generality, we assume the labels of the first l nodes, $\mathbf{y}_1, \ldots, \mathbf{y}_l$, are given (albeit incomplete) and all the other labels are unknown.

Multi-label Learning on Relational Graphs

We first focus on the problem of inferring a more complete label matrix $\hat{\mathbf{Y}} \in \mathbb{R}^{K \times n}$ from a partially observed label matrix $\mathbf{Y}$. (The matrix $\hat{\mathbf{Y}}$ can subsequently be thresholded to yield hard label assignments.) Given a partially observed adjacency matrix $\mathbf{G}$, we can formulate the problem as a classical graph-based semi-supervised learning problem (Zhou and Schölkopf 2004):

$$\underset{\hat{\mathbf{Y}}}{\operatorname{argmin}} \operatorname{Tr}(\hat{\mathbf{Y}}\mathbf{L}\hat{\mathbf{Y}}^\top) + \alpha \left\| \hat{\mathbf{Y}} - \mathbf{Y} \right\|_F^2, \tag{1}$$

where $\mathbf{L} = \mathbf{D} - \mathbf{G}$ is the *graph Laplacian* defined on graph $\mathbf{G}$ and $\mathbf{D}$ is a diagonal matrix with $D_{ii} = \sum_j G_{ij}$. The first term of (1) measures the smoothness of the predicted scores $\hat{\mathbf{Y}}$ on the graph structure and encourages the scores of adjacent nodes to be similar. The second term of (1) penalizes large deviations of the predictions from the given labels (and increases numerical stability). The hyperparameter $\alpha > 0$ adjusts the trade-off between these two objectives.

The optimization problem in (1) can be solved in a closed form as $\hat{\mathbf{Y}} = \alpha \mathbf{Y}(\mathbf{L} + \alpha \mathbf{I})^{-1}$. In general, two different constants of α for trade off can be used: one for the initially labeled nodes and the other for unlabeled nodes. This framework, in combination with alternative choices of the graph Laplacian matrix $\mathbf{L}$, takes several established graph-based semi-supervised algorithms as special cases (up to some scaling factors), *e.g.* the local and global consistent model of Zhou and Schölkopf (2003), and the random-walk with restart (RWR) model of Pan et al. (2004). In this study, we will follow the graph-based semi-supervise learning framework defined in (1) as the centerpiece for MLL.

Link Prediction on Bi-relational Graphs

We now consider the LP problem of inferring missing links in a partially observed graph $\mathbf{G}$. Different from previous work, we also incorporate the known, albeit incomplete, labels $\mathbf{Y}$ in the process.

Constructing a bi-relational graph

We first inject label information by augmenting the original graph $\mathcal{G}$ with additional nodes and edges. Specifically, we introduce K new nodes, one per label. We refer to these newly added nodes as *label nodes* and the original nodes as *data nodes*. In addition, we create a link between the i^{th} data node and the k^{th} label node if node i carries label k, *i.e.* $Y_{ki} = 1$; see box 3 in Figure 1 for a schematic illustration. The resulting bi-relational graph characterizes relationships between two types of nodes, the data and label nodes, and we use $\overline{\mathbf{G}} \in \{0,1\}^{(n+K)\times(n+K)}$ to denote the link matrix of the new graph. $\overline{\mathbf{G}}$ can be decomposed as

$$\overline{\mathbf{G}} = \begin{bmatrix} \mathbf{G} & \mathbf{Y}^\top \\ \mathbf{Y} & \mathbf{H} \end{bmatrix}, \tag{2}$$

where $\mathbf{G}$ is the adjacency matrix of the original graph, $\mathbf{Y}$ encodes the observed association between data nodes and labels (or label nodes), and $\mathbf{H}$, which may be all-zeros or some prior that prescribes the similarities among labels. We use the cosine similarity among observed labels, as suggested before (Wang, Huang, and Ding 2011).

It is important to note that the bi-relational graph $\overline{\mathbf{G}}$ is considered to be only partially observed, because each of its components: $\mathbf{G}$, $\mathbf{Y}$, and $\mathbf{H}$ may be incomplete. We will learn to predict missing links on the augmented bi-relational graph $\overline{\mathbf{G}}$ instead of the original graph $\mathbf{G}$. Adding links between data and label nodes in $\overline{\mathbf{G}}$ is another form for completing $\mathbf{Y}$, complementary to the graph Laplacian based method we introduced in the previous section. It also serves as a key component for linking LP and MLL.

Marginalized denoising model for link prediction

Different from most prior work, we use the recently developed marginalized denoising framework for link prediction (Chen et al. 2012; Chen, Zheng, and Weinberger 2013; Chen and Zhang 2014).

Consider the observed links in an "incomplete" bi-relational graph $\overline{\mathbf{G}}$ as randomly sampled from an (unknown) "complete" bi-relational graph $\overline{\mathbf{G}}^*$. The goal for LP is to find a linear mapping $\mathbf{W}$ to reverse the corruption process and approximately reconstructs the "complete" graph $\overline{\mathbf{G}}^*$ from the "incomplete" graph $\overline{\mathbf{G}}$ such that $\mathbf{W}\overline{\mathbf{G}} \approx \overline{\mathbf{G}}^*$.

Because $\overline{\mathbf{G}}^*$ is unknown, we cannot learn $\mathbf{W}$ directly. Instead, we construct an artificial training set of m further corrupted graphs of $\overline{\mathbf{G}}$, $\widetilde{\mathbf{G}}^1, \ldots, \widetilde{\mathbf{G}}^m$, by randomly removing edges with some probability $p \in [0,1]$. We then train $\mathbf{W}$ to reconstruct $\overline{\mathbf{G}}$ from these further corrupted graphs with

$$\min_{\mathbf{W}} \frac{1}{m} \sum_{j=1}^{m} \|\overline{\mathbf{G}} - \mathbf{W}\widetilde{\mathbf{G}}^j\|_F^2. \tag{3}$$

If this corruption mechanism approximates how $\overline{\mathbf{G}}$ was generated from the "complete" graph $\overline{\mathbf{G}}^*$, re-applying the learned mapping $\mathbf{W}$ to $\overline{\mathbf{G}}$ will approximate the "uncorrupted" graph $\overline{\mathbf{G}}^*$. As far as the learning of $\mathbf{W}$ is concerned,

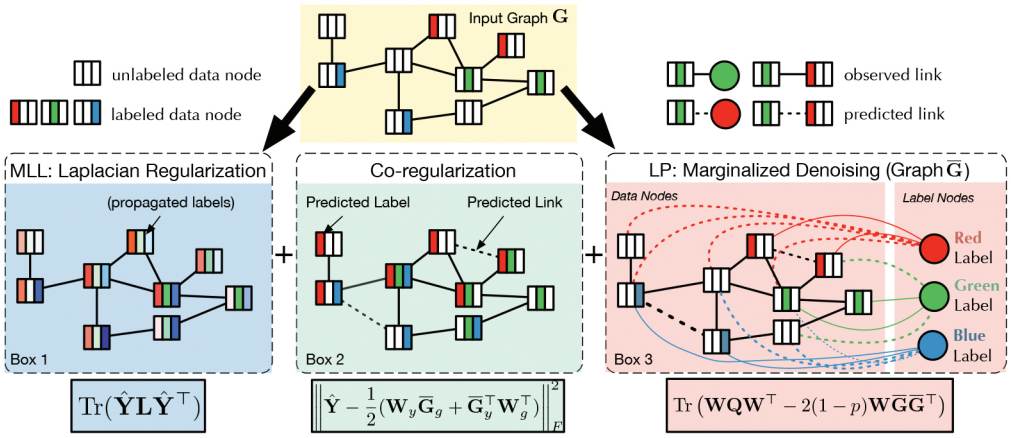

Figure 1: An illustration of MLLP. Each data node can be associated with one or more labels in a label set: Red, Green, and Blue. The original graph is augmented into a bi-relational graph, $\overline{\mathbf{G}}$ (Box 3). The joint model (Box 2) encourages agreement between the predicted labels from MLL (Box 1) and the enriched labels from LP (Box 3).

it is generally better to set m as large as possible. We therefore consider the limiting case, $m \to \infty$, so that (3) becomes

$$\min_{\mathbf{W}} \mathbb{E}\left[\|\overline{\mathbf{G}} - \mathbf{W}\widetilde{\mathbf{G}}\|_F^2\right]_{P(\widetilde{\mathbf{G}}|\overline{\mathbf{G}})}. \tag{4}$$

This expectation is analytically computable in closed form (Chen et al. 2012). It is straightforward to show that (4) can be rewritten as

$$\min_{\mathbf{W}} \mathrm{Tr}(\mathbf{W}\mathbf{Q}\mathbf{W}^\top - 2(1-p)\mathbf{W}\overline{\mathbf{G}}\overline{\mathbf{G}}^\top), \tag{5}$$

where $\mathbf{Q} = (1-p)^2\overline{\mathbf{G}}\overline{\mathbf{G}}^\top + p(1-p)\mathrm{diag}(\overline{\mathbf{G}}\overline{\mathbf{G}}^\top)$. Eq. (5) describes a parabola and its solution can be computed in closed form. Note that the expectation in (4) is still tractable even for more general corruption models (Chen et al. 2012; van der Maaten et al. 2013).

Because of the bi-relational nature of $\overline{\mathbf{G}}$, the mapping $\mathbf{W}$ can be used for both label prediction and link prediction. To simplify notation, let $\mathbf{W} = [\mathbf{W}_g\ \mathbf{W}_y]$, where the subscripts g and y denote the first n and last K *rows* of $\mathbf{W}$ respectively, and we can decompose $\overline{\mathbf{G}} = [\overline{\mathbf{G}}_g^\top\ \overline{\mathbf{G}}_y^\top]^\top$ (the first n and the last K *columns*) accordingly. This approach can be used naturally for both link and label prediction.

Link prediction. By the definition of bi-relational graph $\overline{\mathbf{G}}$ in (2), the original graph $\mathbf{G}$ corresponds to its upper-left corner. We can tease out this part of the interpolated link matrix as

$$\hat{\mathbf{G}}_{\text{dir}} = \mathbf{W}_g\overline{\mathbf{G}}_g, \quad \hat{\mathbf{G}}_{\text{undir}} = \frac{1}{2}(\mathbf{W}_g\overline{\mathbf{G}}_g + \overline{\mathbf{G}}_g^\top\mathbf{W}_g^\top), \tag{6}$$

where $\hat{\mathbf{G}}_{\text{dir}}$ describes the solution in the scenario of directed graphs and $\hat{\mathbf{G}}_{\text{undir}}$ in the scenario of undirected graphs. (In the case of undirected graphs it is advantageous to symmetrize the output after the mapping with $\mathbf{W}$.)

Label prediction. Imputed links between the data and label nodes can be interpreted as predictions of labels. Regardless of the nature of the input graph, we consider these links always undirected and we compute the predicted label matrix, $\hat{\mathbf{Y}}'$, via link prediction as

$$\hat{\mathbf{Y}}' = \frac{1}{2}(\mathbf{W}_y\overline{\mathbf{G}}_g + \overline{\mathbf{G}}_y^\top\mathbf{W}_g^\top). \tag{7}$$

Joint Objective

In this section we combine the LP and MLL problems into a single joint objective, which allows information flow in both directions. The key idea, which forms the cornerstone of this study, is to establish the correspondence between LP and MLL such that these two tasks can exchange information under a unified framework. Note that the second term of the MLL objective (1), $\|\hat{\mathbf{Y}} - \mathbf{Y}\|_F^2$, penalizes large deviations of the learned scores $\hat{\mathbf{Y}}$ from observed labels $\mathbf{Y}$. These observed labels themselves, however, are notoriously incomplete. Instead of relying completely on the original observed labels, we encourage the two approaches, LP and MLL, to agree with each other with respect to their label predictions, *i.e.* we co-regularize the two predictions with the term $\|\hat{\mathbf{Y}} - \hat{\mathbf{Y}}'\|_F^2$. We include the closed form solution for $\hat{\mathbf{Y}}'$, eq. (7), in this co-regularizer and combine it with the two objectives (1) and (5) to obtain the final joint optimization:

$$\underbrace{\mathrm{Tr}\left(\hat{\mathbf{Y}}\mathbf{L}\hat{\mathbf{Y}}^\top\right)}_{\text{prediction smoothness}} + \alpha\underbrace{\left\|\hat{\mathbf{Y}} - \frac{1}{2}(\mathbf{W}_y\overline{\mathbf{G}}_g + \overline{\mathbf{G}}_y^\top\mathbf{W}_g^\top)\right\|_F^2}_{\text{enriched label agreement}} + \beta\underbrace{\mathrm{Tr}(\mathbf{W}\mathbf{Q}\mathbf{W}^\top - 2(1-p)\mathbf{W}\overline{\mathbf{G}}\overline{\mathbf{G}}^\top)}_{\text{link prediction on bi-relational graph}}. \tag{8}$$

This optimization is over $\hat{\mathbf{Y}}$ and $\mathbf{W}$. The constant $\alpha \geq 0$ trades-off the co-regularization terms and $\beta > 0$ balances the two tasks. Note that the last term is crucial for incorporating the original labels $\mathbf{Y}$ in $\overline{\mathbf{G}}$. Without it, the optimization would yield a trivial all-zero solution for $\mathbf{W}$ and $\hat{\mathbf{Y}}$. Figure 1 sketches the intuition behind the joint optimization.

The objective in (8) is jointly convex and quadratic in $\hat{\mathbf{Y}}$ and $\mathbf{W}$. This means that we can obtain the global solution by solving both matrices in closed forms in an alternating fashion. Algorithm 1 summarizes the joint optimization algorithm with detailed alternating update rules. Note that the

most computationally intensive operations, the matrix inversions, conveniently involve only with sparse matrices and are executed out the main loop.

Non-uniform co-regularization. In our formulation in (8), the observed labels only enter the optimization in LP (the third term). To propagate this information to the MLL objective (first term), it makes sense to co-regularize the two methods more for nodes that are accompanied with labels. In practice, we use different α's for nodes with and without labels (α_l, α_u respectively). We have found empirically that it is best to set $\alpha_l \gg \alpha_u$ when there are many unlabeled nodes.

Stacked link prediction. Chen et al. (2012) suggest to improve the performance of marginalized denoising in document representation through stacking. We observe a similar performance boost in our setting. In other words, after solving (8) we "enrich" the graph $\overline{\mathbf{G}}$ with the mapping $\mathbf{W}$ and truncate all entries to be within $[0, 1]$. We then use this enriched link matrix as the input for another round of optimization of (8). We repeat this process until the LP result stops to improve on a hold-out set.

Algorithm 1 Joint link prediction and multi-label learning

Input: Labels $\mathbf{Y}$, Graph $\mathbf{G}$, Parameters $\alpha, \beta > 0$, and level of corruption: $0 < p < 1$

Output: link prediction scores $\hat{\mathbf{G}}$ and label prediction scores $\hat{\mathbf{Y}}$.

Set $H_{ij} = \frac{\mathbf{y}_i^\top \mathbf{y}_j}{||\mathbf{y}_i|| ||\mathbf{y}_i||}$, $\overline{\mathbf{G}} = [\mathbf{G}, \mathbf{Y}^\top; \mathbf{Y}, \mathbf{H}]$, $\mathbf{L} \leftarrow$ graph Laplacian of $\mathbf{G}$, $\mathbf{P} = (1-p)\mathbf{W}\overline{\mathbf{G}}\overline{\mathbf{G}}^\top$. Initialize $\hat{\mathbf{Y}}$ and $\mathbf{W}$ randomly.

$\mathbf{Z}_1 = (\frac{\alpha}{4}\overline{\mathbf{G}}_g\overline{\mathbf{G}}_g^\top + \beta\mathbf{Q})^{-1}$

$\mathbf{Z}_2 = (\frac{\alpha}{4}\overline{\mathbf{G}}_y\overline{\mathbf{G}}_y^\top + \beta\mathbf{Q})^{-1}$

$\mathbf{Z}_3 = (\mathbf{L} + \alpha\mathbf{I})^{-1}$

repeat

$\mathbf{W}_y \leftarrow (\frac{\alpha}{2}(\hat{\mathbf{Y}} - \overline{\mathbf{G}}_y^\top\mathbf{W}_g^\top)\overline{\mathbf{G}}_g^\top + \beta\mathbf{P}_y)\mathbf{Z}_1$

$\mathbf{W}_g \leftarrow (\frac{\alpha}{2}(\hat{\mathbf{Y}}^\top - \overline{\mathbf{G}}_g^\top\mathbf{W}_y^\top)\overline{\mathbf{G}}_y^\top + \beta\mathbf{P}_g)\mathbf{Z}_2$

$\hat{\mathbf{Y}} \leftarrow \frac{\alpha}{2}(\mathbf{W}_y\overline{\mathbf{G}}_g + \overline{\mathbf{G}}_y^\top\mathbf{W}_g^\top)\mathbf{Z}_3$

until convergence

return $\hat{\mathbf{Y}}$ and $\hat{\mathbf{G}}$, as defined in (6)

Experimental Results

We evaluated MLLP, together with several other MLL and LP algorithms on three real-world datasets, two of which are from computational biology and the third is on social networks.

Datasets. 1) *Yeast*: we extracted the yeast PPI graph from HINT (Das and Yu 2012), which is a human-curated high-quality PPI dataset. The functional annotations (labels) of proteins were obtained from the SGD database[1]. We kept 99 function labels that were associated with at least 20 proteins and removed proteins that were not annotated by any of these 99 functions. This resulted in a graph consisting of 2,925 nodes annotated with at least one label.

2) *Human*: the human PPI graph was also taken from HINT. The human proteins were annotated according to the Gene Ontology (Consortium and others 2000). We followed the same filtering steps as in the *Yeast* dataset, obtaining a PPI graph of 4,877 nodes and a set of 267 distinct labels.

3) *LiveJournal*: the whole dataset describes a social network from an online blogging community of more than 3 million users (Yang and Leskovec 2012). We randomly sampled a subgraph of the network consisting of a connected component of 11,921 users. Each user is associated with one or more user-defined groups. Given the observed network and the known groups associated with each user, the LP task is to predict if two users are likely to establish friendship, and the MLL task is to identify hidden user groups that a user is associated with.

Methods. We compared our method *MLLP* to several existing methods that solve the MLL and LP separately. Three state-of-the-art MLL algorithms that are designed to handle incomplete labels, *i.e. MLR-GL* (Bucak, Jin, and Jain 2011), *I-BG* (Wang, Huang, and Ding 2011), and *ProtDM* (Yu et al. 2013), were chosen. We also included three state-of-the-art LP methods: Kernel-based scores (*Kernel*) (Fouss et al. 2012), link propagation (*LinkProp*) (Kashima et al. 2009), and supervised link prediction (*SLP*) (Al Hasan et al. 2006). To demonstrate the benefit of solving MLL and LP jointly, we included the marginalized denoising method (*MDM*) in Eq. (4) on the bi-relational graph, and the graph-based regularization framework (*GR*) in Eq. (1) as the baselines. We also include the alternating method (*Alter*) (Bilgic, Namata, and Getoor 2007) as a baseline. There are numerous choices of how the prediction results can be exchanged between *MDM* and *GR* at each iteration in *Alter*. We choose to add the 100 top-ranked edges predicted by *MDM* to the observed graph for *GR* and add 10 top-ranked instances for each label predicted by *GR* to the known labels for *MDM* because it leads to the best performance. We repeat this procedures for 20 times. We set the hyper-parameters of all methods using 5-fold cross-validation, except the corruption noise p of *MLLP* and *MDM* is set to a high value $p=0.95$, throughout.

Experiment setup and evaluation metrics. Both MLL and LP are essentially tasks to infer missing information in a graph. To establish ground truth for comparing performance of the method considered, we removed labels and edges in three ways. 1) We randomly removed a fraction, $\gamma \in \{0.0, 0.1, 0.2, 0.3, 0.4\}$, of the existing edges. 2) We randomly selected 50% of the nodes and remove *all* label information from them (these are the MLL test nodes). 3) To simulate various degrees of incomplete *training* information, we randomly removed a fraction $\delta \in \{0.0, 0.1, 0.2, 0.3, 0.4\}$ of the label information $\mathbf{Y}$ of all remaining nodes. As there are five choices for γ and δ we ultimately created 25 distinct setups for training. For both MLL and LP, we reported the AUC of the ROC curve (Huang and Ling 2005) of the reconstructed links and induced labels (on the testing nodes) averaged over five runs along with standard deviations.

Performance of link prediction. Fig. 2 summarizes the AUCs of the various methods for link prediction on the three datasets. In the first row, we plot the average AUCs as a function of the fraction of missing links, γ.[2] Here, we fixed the

[1] http://www.yeastgenome.org/

[2] There is no result for $\gamma = 0$, as in this case there is no link to be reconstructed.

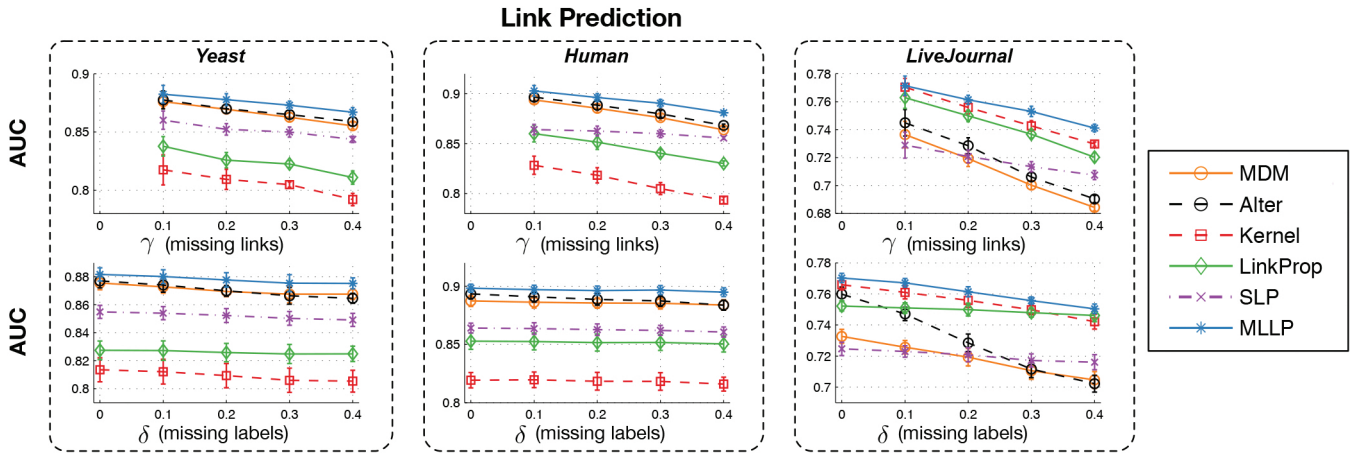

Figure 2: A comparison of various LP algorithms and *MLLP*. The graphs show the averaged AUC scores as a function of missing links γ and missing labels δ. The standard deviation of the AUCs are shown as error bars.

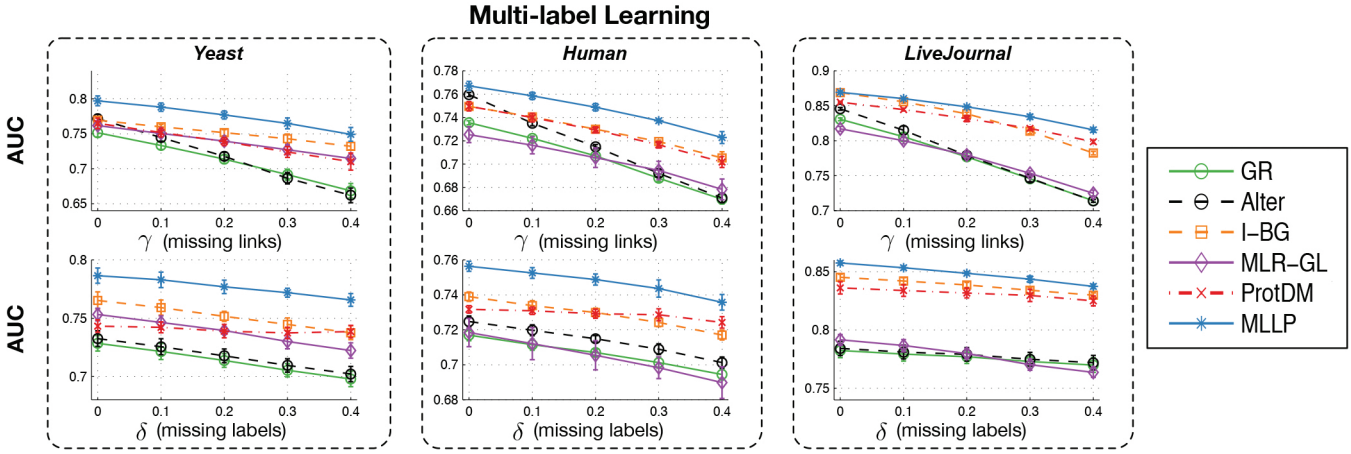

Figure 3: Comparison of various MLL algorithms and *MLLP*. The graphs show the averaged AUC scores as a function of missing links γ and missing labels δ. The standard deviations of the AUCs are shown by error bars.

fraction of missing training labels to $\delta = 0.2$. In the second row, we varied δ and set $\gamma = 0.2$. The baseline model, *MDM*, outperforms the three state-of-the-art methods, *Kernel*, *LinkProp*, and *SLP*, for all settings on the Yeast and Human datasets by a relatively large margin. It indicates that the marginalized denoising model is indeed promising in link prediction. Its performance on the LiveJournal dataset is not satisfactory though. We hypothesize that this may be due to the relatively high sparseness (lower graph density and node degree) of the graph, which makes it difficult to recover the hidden link correlations solely through corruption and denoising. The *Alter* method outperforms *MDM* when the fraction of missing labels (δ) is small, but its performance decreases as δ increases. This is probably because when δ is large, the MLL part, *GR*, is more likely to make wrong predictions and thus makes the bi-relational graph of *MDM* unreliable. Our method *MLLP*, on the other hand, consistently outperforms the other methods on all three datasets. In particular, it boosts the performance of *MDM* significantly on the LiveJournal dataset, which indicates that solving the MLL and LP problems jointly significantly improves the performance of the LP accuracy. As expected, when the fraction of removed links γ increases, the overall LP AUCs decreases. However, removing labels (increased δ) has a less pronounced effect on the LP performance on the Yeast and Human datasets, probably because these two networks are relatively dense enough for the LP algorithms to infer links from the observed links alone.

Performance of multi-label learning Fig. 3 summarizes the MLL performance of various methods following the same setting as in Fig. 2. As shown, the baseline method *GR* for MLL is not as powerful as *MDM* for LP. It performs worse than most of the existing methods. The *Alter* improves upon *GR* in almost all settings. It is worth to note that as the fraction of missing links (γ) increases, the improvement of *Alter* becomes marginal. This is similar to the effect of δ on *Alter* in the LP task; *i.e.* the alternating method is prone to aggregating wrong predictions made by previous iterations when γ is large. Our new method *MLLP* enhanced with the link prediction component, is able to considerably improve its performance. *MLLP* outperforms almost all other methods across different settings except for LiveJournal when

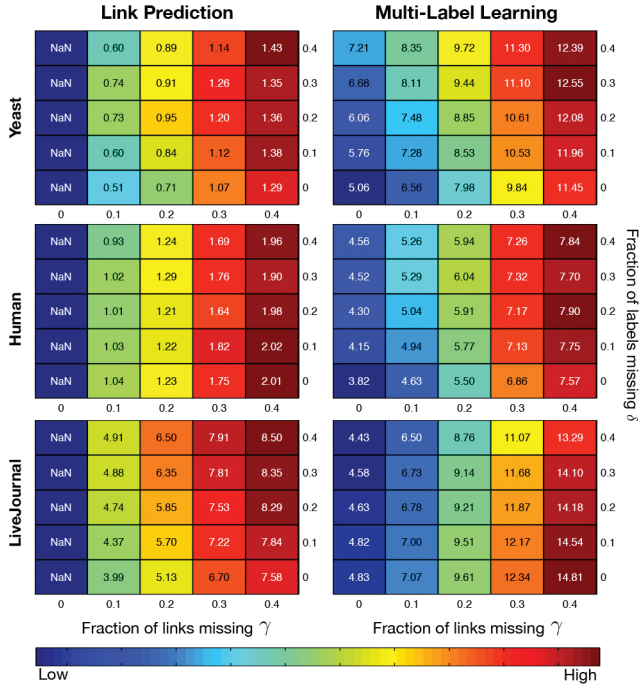

Figure 4: Relative improvements of *MLLP* over the baseline methods *MDM* for LP (upper row) and *GR* for MLL (lower row). The plots show the relative increase in LP AUC (%) and MLL AUC (%) as the fractions of missing labels (x-axis) and missing links (y-axis) varies.

$\gamma = 0$, where it is comparable to *I-BG*. It is worth noting that *MLLP* improves upon the baselines (*MDM* for LP and *GR* for MLL) more pronouncedly as either γ or δ increases.

Comparison of the joint model with the baselines. We define the relative improvement of *MLLP* over the baselines as $\frac{a_1 - a_0}{a_0}$, where a_0 is the AUC score of *MDM* for LP or that of *GR* for MLL, and a_1 is the corresponding AUC score of *MLLP*. Fig. 4 shows the relative improvement in percentage under all 25 settings with varying fractions of missing labels (y-axis) and links (x-axis). A clear trend emerges showing that the relative improvement of *MLLP* upon the baselines increases as the fraction of missing labels and/or links increases. When the fraction of missing links is high, *e.g.* $\gamma = 0.4$, *MLLP* achieves more than 7% boost over *MDM* on the LiveJournal dataset for LP, and more than 10% improvement over *GR* on the Yeast and LiveJournal dataset for MLL. The improvement, especially on the Yeast and Human datasets for LP, is less evident as the fraction of missing labels δ increases. This is consistent with our previous hypothesis that the labels in these two datasets adds relatively less information as the baseline is able to infer links from the observed links alone.

Running time. The time complexity of MLLP is $O(n^3)$ due to matrix inversions and multiplications. The algorithm usually converges within 10 to 20 iterations. On a server with two 8-core Intel Xeon E5-2650@2.60GHz CPUs and 128GB RAM, our MATLAB^{TM} implementation (accelerated with an NVIDIA K40 GPU) required about 170s for a run of MLLP until convergence on the *LiveJournal* dataset and around 23s on the *Human* dataset (without GPU 7m and 1m respectively). For perfect reproducibility, we provide the source code and data sets online at http://www.cse.wustl.edu/~chenz/codes/mllp.tar.gz.

Related Works and Discussion

Link prediction. The existing works on LP can be roughly grouped into three categories. 1) Similarity based methods, *e.g.* Adamic-Adar Index (Adamic and Adar 2003), extract similarity scores between nodes based on graph topologies. The scores are later thresholded or ranked to form links. 2) Latent factor models explain the observed adjacency matrix using a small number of latent factors. The latent factors are usually learned through matrix factorization (Koren, Bell, and Volinsky 2009) or Bayesian models (Miller, Griffiths, and Jordan 2009). 3) Supervised methods treat LP as binary classification problems. MLLP for LP can be considered as a special instance of supervised learning methods, where we learn the similarity scores using a supervised learning scheme trained on conceptually "infinitely many" training examples. Our work may be most similar to the work (Chen and Zhang 2014), who already explore marginalized denoising for LP, however do not incorporate MLL.

Multi-label learning. Many existing works on MLL explicitly incorporate graph structure into models such as the traditional graph-based semi-supervised models (Zhou et al. 2003) and collective classification methods (Kong, Shi, and Philip 2011). A few existing work explicitly addresses the incompleteness of the class labels though. MLR-GL (Bucak, Jin, and Jain 2011) extends the multi-label ranking approaches with group lasso technique to reduce the penalty on possible unassigned labels. I-BG (Wang, Huang, and Ding 2011) constructs a similar bi-relational graph but uses Random Walk with Restart (RWR) model to learn the relevance between a label and a data nodes. MLL can also be cast as a hyperedge prediction problem over hypergraphs (Sun, Ji, and Ye 2008). In fact, the constructed bi-relational graph can be viewed as a dyadic version of a hypergraph, where a hyperedge is used to connect a label node with multiple data nodes. Using the dyadic edges, we are able to perform MLL and LP jointly.

Joint link prediction and multi-label learning. Although a collection of methods have been developed to solve the LP and MLL problems *independently*, little effort has been devoted to solving the two problems jointly. There are works that have alluded to the close connection between the two problems. For example, the graph Laplacian (Zhou and Schölkopf 2004; Fouss et al. 2012) and variants of graph kernels serve as the main building blocks for many similarity-based LP and MLL methods. Bilgic et. al. (2007) proposed to alternately apply LP and MLL algorithms and showed that it can improve the performance of both tasks. However, it is difficult to prevent unreliable predictions being exchanged from one task to the other. This problem is further exacerbated by missing links and labels as shown in our experiment. MLLP, on the other hand, combines LP and MLL seamlessly into a joint optimization and enables both tasks to benefit from each other.

Acknowledgments

ZC and WZ are supported in part by Natural Science Foundation of China (31300999), the municipal government of Wuhan, Hubei, China (2014070504020241 and the Talent Development Program), and an internal research grant of Jianghan University, Wuhan, China, as well as by United States National Institutes of Health (R01GM100364). KQW is supported by NSF grants IIS-1149882, EFRI-1137211. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

Adamic, L., and Adar, E. 2003. Friends and neighbors on the web. *Social networks* 25(3):211–230.

Al Hasan, M.; Chaoji, V.; Salem, S.; and Zaki, M. 2006. Link prediction using supervised learning. In *SDM06: Workshop on Link Analysis, Counter-terrorism and Security*.

Bilgic, M.; Namata, G. M.; and Getoor, L. 2007. Combining collective classification and link prediction. In *Data Mining Workshops, 2007. ICDM Workshops 2007. Seventh IEEE International Conference on*, 381–386. IEEE.

Bucak, S.; Jin, R.; and Jain, A. 2011. Multi-label learning with incomplete class assignments. In *CVPR*, 2801–2808. IEEE.

Chen, Z., and Zhang, W. 2014. A marginalized denoising method for link prediction in relational data. In *SDM*.

Chen, M.; Xu, Z.; Weinberger, K.; and Sha, F. 2012. Marginalized denoising autoencoders for domain adaptation. In *ICML*. ACM. 767–774.

Chen, M.; Zheng, A.; and Weinberger, K. 2013. Fast image tagging. In *ICML*, volume 28, 1274–1282.

Consortium, G. O., et al. 2000. Gene ontology: tool for the unification of biology. *Nature genetics* 25(1):25–29.

Das, J., and Yu, H. 2012. Hint: High-quality protein interactomes and their applications in understanding human disease. *BMC systems biology* 6:92.

Fouss, F.; Francoisse, K.; Yen, L.; Pirotte, A.; and Saerens, M. 2012. An experimental investigation of kernels on graphs for collaborative recommendation and semisupervised classification. *Neural networks* 31:53–72.

Huang, J., and Ling, C. X. 2005. Using auc and accuracy in evaluating learning algorithms. *Knowledge and Data Engineering, IEEE Transactions on* 17(3):299–310.

Jansen, R.; Yu, H.; Greenbaum, D.; Kluger, Y.; Krogan, N. J.; Chung, S.; Emili, A.; Snyder, M.; Greenblatt, J. F.; and Gerstein, M. 2003. A bayesian networks approach for predicting protein-protein interactions from genomic data. *Science* 302(5644):449–453.

Kashima, H.; Kato, T.; Yamanishi, Y.; Sugiyama, M.; and Tsuda, K. 2009. Link propagation: A fast semi-supervised learning algorithm for link prediction. In *SDM*, volume 9, 1099–1110. SIAM.

Kong, X.; Shi, X.; and Philip, S. Y. 2011. Multi-label collective classification. In *SDM*, volume 11, 618–629. SIAM.

Koren, Y.; Bell, R.; and Volinsky, C. 2009. Matrix factorization techniques for recommender systems. *Computer* 42(8):30–37.

Kourmpetis, Y.; van Dijk, A.; Bink, M.; van Ham, R.; and Ter Braak, C. 2010. Bayesian markov random field analysis for protein function prediction based on network data. *PloS one* 5(2).

Liben-Nowell, D., and Kleinberg, J. 2007. The link-prediction problem for social networks. *Journal of the American society for information science and technology* 58(7):1019–1031.

Miller, K.; Griffiths, T.; and Jordan, M. 2009. Nonparametric latent feature models for link prediction. In *NIPS*, volume 9, 1276–1284.

Pan, J.; Yang, H.; Faloutsos, C.; and Duygulu, P. 2004. Automatic multimedia cross-modal correlation discovery. In *KDD*, 653–658. ACM.

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

van der Maaten, L.; Chen, M.; Tyree, S.; and Weinberger, K. 2013. Learning with marginalized corrupted features. In *ICML*, volume 28, 410–418.

Vazquez, A.; Flammini, A.; Maritan, A.; and Vespignani, A. 2003. Global protein function prediction from protein-protein interaction networks. *Nature biotechnology* 21(6):697–700.

Wang, H.; Huang, H.; and Ding, C. 2011. Image annotation using bi-relational graph of images and semantic labels. In *CVPR*, 793–800. IEEE.

Yang, J., and Leskovec, J. 2012. Defining and evaluating network communities based on ground-truth. In *ICDM*, 745–754.

Yu, G.; Domeniconi, C.; Rangwala, H.; and Zhang, G. 2013. Protein function prediction using dependence maximization. In *Machine Learning and Knowledge Discovery in Databases*. Springer. 574–589.

Zhou, D., and Schölkopf, B. 2004. A regularization framework for learning from graph data. In *ICML 2004 Workshop on Statistical Relational Learning and its Connections to Other Fields*, 132.

Zhou, D.; Bousquet, O.; Lal, T. N.; Weston, J.; and Schölkopf, B. 2003. Learning with local and global consistency. In *NIPS*, volume 16, 321–328.