Robust Semi-Supervised Learning on Multiple Networks with Noise

Junting Ye[1] and Leman Akoglu[2]

[1]Department of Computer Science, Stony Brook University
[2]Heinz College of Information Systems and Public Policy, Carnegie Mellon University
juyye@cs.stonybrook.edu, lakoglu@cs.cmu.edu

Abstract. Graph-regularized semi-supervised learning has been effectively used for classification when (i) data instances are connected through a graph, and (ii) labeled data is scarce. Leveraging multiple relations (or graphs) between the instances can improve the prediction performance, however noisy and/or irrelevant relations may deteriorate the performance. As a result, an effective weighing scheme needs to be put in place for robustness.
In this paper, we propose iMUNE, a **robust** and effective approach for multi-relational graph-regularized semi-supervised classification, that is immune to noise. Under a **convex formulation**, we infer weights for the multiple graphs as well as a solution (i.e., labeling). We provide a careful analysis of the inferred weights, based on which we devise an algorithm that filters out irrelevant and noisy graphs and produces weights proportional to the informativeness of the remaining graphs. Moreover, iMUNE is **linearly** scalable w.r.t. the number of edges. Through extensive experiments on various real-world datasets, we show the **effectiveness** of our method, which yields superior results under different noise models, and under increasing number of noisy graphs and intensity of noise, as compared to a list of baselines and state-of-the-art approaches.

1 Introduction

Given *(i)* a network with *multiple* different relations between its nodes, and *(ii)* labels for a small set of nodes, how can we predict the labels of the unlabeled nodes in a *robust* fashion? Robustness is a key element especially when the data comes from sources with varying veracity, where some relations may be irrelevant or noisy for the prediction task.

This abstraction admits various real-world applications. For example, in fraud detection one may try to classify individuals as fraudulent or not based on the phone-call, SMS, etc. interactions. In biology, genes are classified as whether or not they perform a certain function through various similarity and interaction relations between them.

Accomplishing the above task requires addressing two main problems: (1) identifying and filtering out irrelevant and noisy relations, and (2) automatically weighing other relations by their informativeness for the task. Existing methods either are vastly affected in the presence of noise [1], produce locally optimal solutions due to their non-convex objective formulations [2–4], use only the labeled data [5–7], or are too expensive to compute [8–11].

In this work we introduce iMUNE, a robust, scalable, and effective graph-regularized semi-supervised classification approach for MUlti-relational NEtworks. In the example

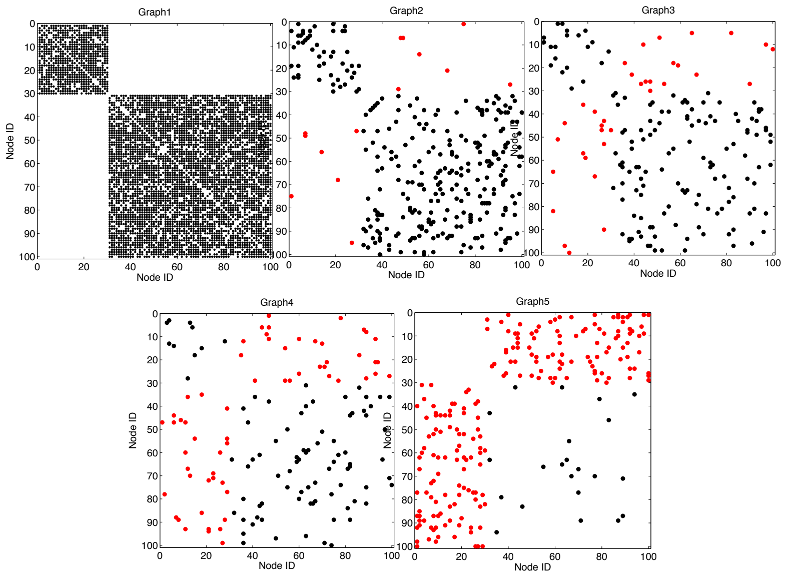


Fig. 1: A synthetic multi-relational graph ($n = 100$ nodes, $m = 5$ views), with 3 informative (top) and 2 noisy (bottom) graphs. Shown are adjacency matrices; red dots: cross-edges between nodes from two different classes, black dots: within-class edges. G_1–G_3 are in order of informativeness. G_4 depicts random noise. G_5 contains adversarial noise. Inferred weights (all graphs): $[25.17, 16.54, 12.79, 17.82, 27.68]$, Average Precision (AP) = 0.734. Weights after noisy graphs removed: $[0.5000, 0.3003, 0.1997, 0, 0]$, AP = 0.974.

shown in Fig. 1, iMUNE recognizes and removes G_4 and G_5 as irrelevant/noisy, and estimates weights for relations G_1-G_3 so as to combine them effectively to achieve improved performance. Our contributions are as follows.

- **Model formulation:** Under a convex formulation, we simultaneously estimate weights for the multiple relations (also graphs or views) as well as a solution (labeling) that utilizes a weighted combination of them. (Sec. 2)
- **Analysis of weights:** We show that in the presence of noise, the inferred weights reflect the impact of different relations on the solution, where both dense informative and irrelevant/noisy graphs receive large weights. (Sec. 2.3)
- **Robust algorithm:** Analysis of weights enable us to devise a robust algorithm that filters out irrelevant/noisy graphs, so as to produce weights proportional to the informativeness of graphs and yield improved performance. (Sec. 3)
- **Scalability:** Our proposed approach scales linearly w.r.t. the number of edges in the combined graph. (Sec. 3.1)
- **Effectiveness:** We show the efficacy of iMUNE on real-world multi-networks with (i) varying number of relevant/noisy graphs, (ii) under different noise models, and (iii) varying intensity of noise; where it outperforms six baseline approaches including the state-of-the-art. (Sec. 4)
- **Reproducibility:** We share the code of iMUNE and all datasets in experiments at `http://www3.cs.stonybrook.edu/%7ejuyye/semi/semi.html`.

2 Problem Formulation

In this work we consider real-world problem settings in which (1) the problem is cast as a binary classification task, (2) data objects are related through multiple different relationships, and (3) ground-truth class labels are scarce. The data can be represented as a *multi-graph*, in which the nodes represent data objects and multiple sets of undirected edges capture associations implied by different relationships.

Using various relationships between data objects may provide more information for a given classification task, especially when input labels are scarce. Collectively, more accurate predictions can be made by combining these *multiple association networks*. However, it is not realistic to assume that all available relationships (i.e., graphs) would be equally, if at all, relevant for a given prediction task. Filtering irrelevant/intrusive relations is especially important when the data sources cannot be carefully controlled—for example, when data is collected from various repositories with varying veracity. In addition, the input graphs may have varying degree of relevance for a task, which necessitates a careful weighing scheme.

Overall, it is essential to build robust classification models that can effectively leverage multiple relationships by carefully weighing relevant graphs while filtering out the intrusive ones. Our work addresses this problem of Robust Semi-supervised Classification for MUlti-NEtworks (RSC-MUNE): Given a binary classification task, a multi-graph, and a small set of labeled objects, the goal is to build an effective classifier that is *robust to noisy and irrelevant data*. We give the formal problem definitions as follows.

Definition 1. MULTI-GRAPH: *A multi-relational graph (or a multi-graph) $\mathcal{G}(V,\mathcal{E})$ consists of a set of graphs (or relations) $\{G_1(V,E_1),\ G_2(V,E_2),\ \ldots,\ G_m(V,E_m)\}$, on the same node set V, $|V| = n$. Undirected (weighted) edges $\mathcal{E} = \{E_1,\ldots,E_m\}$ correspond to links implied by m different types of relations, where we denote $|\mathcal{G}| = m$.*

Definition 2. RSC-MUNE PROBLEM: *Given a multi-graph $\mathcal{G}(V,\mathcal{E})$, $|\mathcal{G}| = m$, and a set of labeled seed nodes $L \subset V$; devise a learning procedure to infer the labels of unlabeled nodes $V \backslash L$, which assigns a list of weights $\mathbf{w} = \{w_1,\ldots,w_m\}$ to individual graphs such that (i) intrusive graphs are filtered (i.e., $w_k = 0$), and (ii) relevant graphs receive weights relative to their informativeness.*

2.1 Graph-based Semi-supervised Learning

There exist various objective formulations for graph-regularized semi-supervised classification provided a *single* graph [13–17]. Generalizing from those traditional semi-supervised learning objectives to multi-graphs, we can write

$$\begin{aligned} \arg\min_{\mathbf{f},\mathbf{w}}\ \ & \|\mathbf{f}-\mathbf{y}\|_2^2 + \lambda \sum_k \mathbf{f}^\top w_k \mathbf{L_k} \mathbf{f} \\ s.t.\ \ & w_k \geq 0,\ \sum_k w_k = 1 \end{aligned} \tag{1}$$

where λ is a regularization parameter, $\boldsymbol{L_k}$ is the normalized Laplacian matrix of kth graph, w_k is the weight of $\boldsymbol{L_k}$, $\mathbf{y}$ is the input vector of known labels and $\mathbf{f}$ is the solution.

This objective function, however, is non-convex in both $\mathbf{f}$ and $\mathbf{w}$. To get around this, several previous approaches have proposed alternating optimization schemes for similar objectives [2, 4]. However, these methods only produce locally optimal solutions.

2.2 Objective Formulation

In this work, inspired by the *TSS* approach [1], we introduce a scheme that infers $\mathbf{f}$ and $\mathbf{w}$ *together* under a *convex* setup. The graph weights we infer (i.e., w_k's) capture the *impact* that each graph has on the solution $\mathbf{f}$. Building on this interpretation, we devise a learning procedure that estimates $\mathbf{f}$ which is *robust* to intrusive graphs.

Our objective function is defined as in Equ. (2).

$$\begin{aligned} &\min_{\mathbf{f},\xi} \ (\mathbf{f}-\mathbf{y})^\top(\mathbf{f}-\mathbf{y}) + c_0\sum_{k=1}^{m}\xi_k \\ &\qquad s.t.\ \ \mathbf{f}^\top\mathbf{L_k}\mathbf{f} \le c+\xi_k,\ \xi_k \ge 0\ ,\forall k=1,\dots,m \end{aligned} \tag{2}$$

The dual form of Equ. (2) that estimates the graph weights as well as the final solution are respectively given in Equ. (3) and (4) (derivations are omitted for brevity).

$$\begin{aligned} &\min_{\mathbf{w}} \ \mathbf{y}^\top(\mathbf{I}+\sum_k w_k\mathbf{L_k})^{-1}\mathbf{y} + c\|\mathbf{w}\|_1 \\ &\quad s.t.\ \ c_0 \ge w_k \ge 0\ ,\forall k=1,\dots,m \end{aligned} \tag{3}$$

$$\mathbf{f} = (\mathbf{I}+\sum_k w_k\mathbf{L_k})^{-1}\mathbf{y} \tag{4}$$

Handling class bias In semi-supervised learning, only part of the nodes are labeled for training, and the rest are unlabeled (depicted with '0'). For each node type ('+1','0', '−1'), we assign a different penalty coefficient, c_+, c_u, c_- respectively. Let $\mathbf{C}$ be a $n\times n$ diagonal matrix, called the *class penalty matrix*, where $\mathbf{C}(i,i) = c_+$ if $y_i = 1$, c_- if $y_i = -1$, and c_u if $y_i = 0$. As such, the criterion in Equ. (2) can be reformulated:

$$\min_{\mathbf{f},\xi}(\mathbf{f}-\mathbf{y})^\top\mathbf{C}(\mathbf{f}-\mathbf{y}) + c_0\sum_{k=1}^{m}\xi_k$$

The dual form and the solution are Equ. (5) and (6).

$$\begin{aligned} &\min_{\mathbf{w}} \ \mathbf{y}^\top\mathbf{C}(\mathbf{C}-\sum_k w_k\mathbf{L_k})^{-1}\mathbf{C}\mathbf{y} + c\|\mathbf{w}\|_1 \\ &\quad s.t.\ \ c_0 \ge w_k \ge 0\ ,\forall k=1,\dots,m \end{aligned} \tag{5}$$

$$\mathbf{f}^* = (\mathbf{C}+\sum_k w_k\mathbf{L_k})^{-1}\mathbf{C}\mathbf{y}\ . \tag{6}$$

The dual program in Equ. (5) is convex and can be solved (e.g., using the projected gradient descent method) to infer the graph weights $\mathbf{w}$. One can then plug in those weights directly into Equ. (6) to estimate $\mathbf{f}^*$. However, this procedure as we show in the experiments yields inferior results in the presence of irrelevant and noisy graphs.

2.3 Graph Weights Interpreted

Next we provide a detailed discussion on the interpretation of the inferred weights by Equ. (3) (instead of Equ. (5) for brevity). In a nutshell, we show that in the presence of intrusive graphs, the weights do *not* reflect the relative *informativeness* of individual graphs—but rather the relative *impact* of each graph on the solution.

Ideally, we want to infer a weight w_k for each graph G_k proportional to its informativeness for the task, where the weights for intrusive graphs are zero. For example, in Fig. 1 we illustrate a toy multi-graph with five views. The ideal weights would be $w_1 > w_2 > w_3 > w_4 = w_5 = 0$. As we show in the following, however, the estimated weights should be interpreted carefully when we have intrusive graphs.

G_k's with larger $\mathbf{f}'\mathbf{L_k}\mathbf{f}$ tend to get larger w_k We have the dual problem $d(\mathbf{w})$ in (3) when learning the weights. We know from basic calculus that

$$\frac{\partial}{\partial x}Y^{-1} = -Y^{-1}(\frac{\partial}{\partial x}Y)Y^{-1} \,. \tag{7}$$

Thus we derive the derivative of $d(\mathbf{w})$ w.r.t w_k as

$$\frac{\partial d(\mathbf{w})}{\partial w_k} = -\mathbf{y}^\top(\mathbf{I} + \sum_{i=1}^{m} w_i\mathbf{L_i})^{-1}\mathbf{L_k}(\mathbf{I} + \sum_{i=1}^{m} w_i\mathbf{L_i})^{-1}\mathbf{y} + c \tag{8}$$

Since $\mathbf{f} = (\mathbf{I} + \sum_{i=1}^{m} w_i\mathbf{L_i})^{-1}\mathbf{y}$, we obtain

$$\frac{\partial d(\mathbf{w})}{\partial w_k} = -\mathbf{f}^\top\mathbf{L_k}\mathbf{f} + c \tag{9}$$

Based on (9), we make the following inference:

Both dense informative and intrusive graphs G_k has large $\mathbf{f}^\top\mathbf{L_k}\mathbf{f}$—and hence large w_k Consider a graph with no noisy edges (i.e., no edges between nodes from different classes) but with high edge density among nodes that belong to the same class. For such a graph, $\mathbf{f}^\top\mathbf{L_k}\mathbf{f} = \sum_{i,j\in V}\mathbf{W_k}(i,j)(\frac{f_i}{\sqrt{\mathbf{D_k}(i,i)}} - \frac{f_j}{\sqrt{\mathbf{D_k}(j,j)}})^2$ can be large due to the numerous non-zero (although likely small) quadratic terms in the sum. Importantly, it is not only the dense informative graphs that would have large $\mathbf{f}^\top\mathbf{L_k}\mathbf{f}$, but *also the intrusive graphs*. This is due to the many cross-edges that irrelevant and noisy graphs have between nodes from different classes, that would yield large quadratic terms. We demonstrate this through the inferred weights on our example multi-graph in Fig. 1. Notice that while the highly informative G_1 and G_2 receive large weight, the noisy graphs G_4 and G_5 also obtain comparably large weights.

3 iMUNE Algorithm

Our goal is to filter out the intrusive graphs. The main idea is to explore the search space through simulated annealing by carefully removing large-weighted graphs one at

Algorithm 1 iMUNE (proposed algorithm for robust semi-supervised classification for multi-graphs with noise)

```
Input: Multi-graph $\mathcal{G} = \{G_1, \ldots, G_m\}$, labeled nodes $L$, initial temperature $t$, class penalty matrix $\mathbf{C}$
Output: Label estimations $\mathbf{f}$
 1: Init $\mathbf{y}$ with $L$;
 2: $best\mathbf{f} \leftarrow \emptyset$, $bestP = 0$, $m = |\mathcal{G}|$, $Q \leftarrow \mathcal{G}$
 3: while $Q$ is not empty do
 4:   $\mathcal{GS} \leftarrow dequeue(Q)$
 5:   $cvP =$ Compute cross validation performance of $\mathcal{GS}$
 6:   if $rand(0,1) \leq \exp(\frac{cvP-bestP}{t^{m-|\mathcal{GS}|+1}})$ then
 7:     $\mathbf{w}_{\mathcal{GS}} \leftarrow$ Solve (5) using $\mathcal{GS}$ and input $\mathbf{C}$
 8:     $\mathbf{f}_{\mathcal{GS}} \leftarrow$ Compute solution using (6) and $\mathbf{w}_{\mathcal{GS}}$
 9:     Cluster the weights: $(W_s, W_l) \leftarrow 2\text{-}means(\mathbf{w}_{\mathcal{GS}})$
10:     for each $G_k \in \mathcal{GS}$ for which $w_k \in W_l$ do
11:       $v \leftarrow hash(\mathcal{GS} \backslash G_k)$
12:       if $v$ is null then $Q \leftarrow Q \cup \mathcal{GS} \backslash G_k$
13:     end for
14:     if $cvP > bestP$ then $best\mathbf{f} \leftarrow \mathbf{f}_{\mathcal{GS}}$, $bestP = cvP$
15:   end if
16: end while
17: return $best\mathbf{f}$
```

a time. Steps of our proposed algorithm is outline in Algorithm 1. We start with introducing a queue of graph-sets, which initially includes the set of all graphs (line 2). We process the graph-sets in the queue one by one until the queue becomes empty (line 3). For each graph-set $\mathcal{GS}$ that we dequeue (line 4), we compute its cross-validation performance cvP on the labeled data (line 5). In our experiments, we use average-precision (AP) as our performance metric. This metric is more meaningful than accuracy, especially in the face of class bias.

We record the best AP as $bestP$ during the course of our search (line 14). With probability $\exp(\frac{cvP-bestP}{t^{m-|\mathcal{GS}|+1}})$, we "process" the graph-set in hand (lines 7-13, which we will describe shortly), otherwise we discard it. In line 6, $t \leq 1$ is the temperature parameter of simulated annealing and $(m - |\mathcal{GS}|)$ denotes the number of removed graphs from the original set. If the graph-set $\mathcal{GS}$ in hand yields a cvP that is larger than $bestP$, we always process the set further, since when $(cvP - bestP) \geq 0$, $\exp(\frac{cvP-bestP}{t^{m-|\mathcal{GS}|+1}}) \geq 1$. On the other hand, if $\mathcal{GS}$ yields inferior performance, we still process it with some probability that is proportional to the size of the graph-set. That is, the probability of processing a set decreases as they have more graphs removed from the original set. The probability is also inversely proportional to the performance distance $(cvP - bestP)$. The larger the gap, the higher the chance that $\mathcal{GS}$ will be discarded.

Next we describe the steps to "process" a graph-set $\mathcal{GS}$. We first solve the optimization problem (5) using $\mathcal{GS}$ for the graph weights $\mathbf{w}_{\mathcal{GS}}$ and compute the solution using $\mathbf{w}_{\mathcal{GS}}$ in (6) (lines 7-8). Next we cluster the weights into two groups, those with small weights W_s and those with large weights W_l (line 9). We know, through the analysis in §2.3, that intrusive graphs are *among* the large-weighted graphs. The issue is we do not know in advance which ones, as dense informative ones are likely to also belong to this group. As such, we create from $\mathcal{GS}$ candidate graph-sets that contain all but each large-weighted graph and add those to the queue. Note that we maintain a hash table of the candidate graph-sets (line 11), so that we avoid re-considering the same sets that might be generated through different removal paths. At the end, we return the solution $best\mathbf{f}$ with the $bestP$.

3.1 Complexity Analysis

At each node of our "search tree", we solve Equ. (5) using projected gradient descent, where the main computation involves computing the gradient (See Equ. (8) in §2.3). The gradient involves the term $(\mathbf{I} + \sum_{i=1}^{m} w_i \mathbf{L_i})^{-1}$, i.e., the inverse of a $(n \times n)$ matrix which is $O(n^3)$ if done naively. The same is true for the solution $\mathbf{f}$ which requires a similar inverse operation (See Equ. (4) or Equ. (6)). Importantly, however, we do not compute the inverse explicitly, because it always appears in vector form $\mathbf{x} = (\mathbf{I} + \sum_{i=1}^{m} w_i \mathbf{L_i})^{-1}\mathbf{y}$. We can obtain $\mathbf{x}$ as a solution of sparse linear systems [18], where the computational cost of the derivative is linear w.r.t. the number of non-zero entries of $\sum_{i=1}^{m} w_i \mathbf{L_i}$, i.e., proportional to the number of edges in the multi-graph.

Computing the dual objective then takes $O(s|\mathcal{E}|)$ for s number of gradient steps. All in all, total time complexity of an implementation that traverses each search path in parallel is $O(s|\mathcal{E}|m_u)$, which is linear on the total number of edges in the multi-graphs with small m_u (max. number of noisy graphs) and constant s.

4 Evaluation

4.1 Experiment Setup

Datasets. The multi-graphs used in our work are publicly available, and are listed in Table 1. *(i) RealityMining* [19] is a dataset collected through tracking activities on cellphones. It contains 4 different relations between two classes (MIT Sloan and CS students): phone call, SMS, friendship, and Bluetooth scans that capture proximity relations. *(ii) Protein* [1] consists of Yeast proteins, associated through 5 different relations. Those proteins with function *transport facilitation* constitute the positive class, and others are negative. *Gene1* and *Gene2* contain different sets of Yeast genes, each associated through 15 different genomic sources. The genes are labeled according to *Gene Ontology association* file from the *Saccharomyces Genome Database*. For *Gene1*, we choose the label with the maximum number of genes in *Cellular Component* (CC) domain as

Table 1: Four real-world multi-graph datasets.

Dataset	#Graphs	#Nodes	#Pos.	#Neg.
RealityMining	4	78	27	51
Protein	5	3,588	306	3,282
Gene1	15	1,724	185	1,539
Gene2	15	3,146	214	2,932

positive class. We construct *Gene2* in a similar way, where this time genes in *Molecular Function* (MF) domain are labeled as positive. See [5] for more details on datasets.

Baselines. We compare iMUNE against four state-of-the-art: *ClusDCA* [20], *TSS* [1], *RobustLP* [2], and *GeneMania* [5]. We also introduce two simple baselines, *EqlWght* that assigns equal weight to all graphs and *PerfWght* that assigns weights proportional to the cross-validation accuracy of individual graphs on labeled nodes. To make it a fair game, we use the same class-bias penalties described in §4.2 for the compared methods.

Noise-testing. To test the robustness of the methods, we injected intrusive graphs with varying level, model, and intensity of noise as described below.

- *Number of intrusive graphs*: We tested the effect of increasing noise level on classification performance by injecting 2, 4 and 6 intrusive graphs at a time.
- *Noisy graph models*: We adopted 3 strategies to generate intrusive graphs; (1) Erdos-Renyi random graphs (ER), (2) edge-rewired original graphs (RW), and what we call (3) adversarial graphs (AV) (where most edges are cross-edges between the different classes).
- *Noise intensity* (Low/High): Intensity reflects injected graph density (L: 5%, H: 50%) for ER, ratio of within-class edges randomly rewired to become cross-edges (L: 60%, H: 80%) for RW, and ratio of cross-edges (L: 60%, H: 80%) for AV model.

Overall, there are 3 different number of injected graphs, 3 noise models, and 2 noise intensities. Overall, the "noise-testing" involves 18 (3*3*2) different settings.

4.2 Parameters

Our algorithm expects two hyper parameters; the initial simulated annealing temperature t, and the class penalty matrix $\mathbf{C}$. We describe how we set these in the following. Note that our objective function in Equ. (5) has two further (hyper) parameters c and c_0, which are chosen by cross-validation.

Initial temperature t. As we remove more and more graphs from the input multi-graph, the probability of further considering a set with inferior performance should decrease. That is when $d = (cvP - bestP) < 0$, $p = \exp(\frac{d}{t^{m-|\mathcal{GS}|+1}})$ should decrease as $r = (m - |\mathcal{GS}| + 1)$ increases. As such, we need $t \leq 1$. Assume that we have an expected range $[m_l, m_u]$ for the number of intrusive graphs in the data where m_l and m_u respectively denote the minimum and maximum number. We would then want the probability $p = \exp(\frac{d}{t^r})$ to approach zero as r gets closer to m_u even for a considerably small d. That is, as $r \to m_u$ and $0 > d \geq d_{min}$ for small d_{min}, we want $p_{max} > p > 0$ for small p_{max}. Since $t = (\frac{d}{\ln p})^{\frac{1}{r}}$, the range for t satisfying the above constraints can be given as $t \in [(\frac{d_{min}}{\ln p_{max}})^{\frac{1}{m_u}}, (\frac{d_{min}}{\ln p_{max}})^{\frac{1}{m_l}}]$. Empirically, we let $d_{min} = -0.1$ and $p_{max} = 0.01$. For example, if we expect $m_l = 5$ and $m_u = 10$, then the initial temperature is chosen randomly from $t \in [0.465, 0.682]$.

Class penalty matrix $\mathbf{C}$. As described in §2.2, we can normalize biased class distribution by assigning larger penalty to minority-class ('$+1$') mis-classification. Recall that c_+, c_u, c_- denote penalty coefficients for classes '$+1$', '0' (unlabeled), and '-1', respectively. We set these parameters as $c_+ = 1 + sign(1 - 2p) * \gamma * \max(p, 1 - p)$,

Table 2: *i*MUNE *consistently outperforms competing methods across various real-world and synthetic datasets*. Dataset *RM* is injected with 2, 4, and 6 intrusive graphs with various noise settings. Values depict mean Average Precision (10 runs).

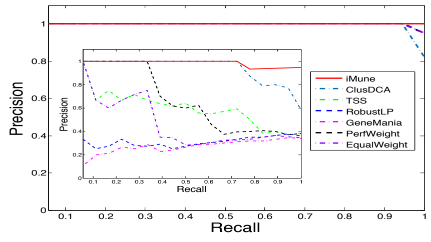
Dataset	#Graph	Model	Intensity	iMUNE	*PerfWght*	*EqlWght*	*TSS*	*RobustLP*	*Mania*	*ClusDCA*
	4	——	——	**0.970**	0.944	0.939	**0.970**	0.947	0.951	0.933
	4+2	AV	Low	**0.970**	0.707	0.554	0.525	0.851	0.470	0.894
	4+2	AV	High	**0.970**	0.611	0.484	0.554	0.809	0.359	0.898
	4+2	RW	Low	**0.970**	0.695	0.669	0.718	0.873	0.505	0.912
	4+2	RW	High	**0.970**	0.563	0.537	0.65	0.824	0.290	0.927
	4+2	ER	Low	**0.970**	0.905	0.841	0.657	0.928	0.773	0.918
	4+2	ER	High	**0.970**	0.942	0.920	0.895	0.930	0.883	0.936
	4+4	AV	Low	**0.970**	0.531	0.372	0.390	0.427	0.260	0.866
RM	4+4	AV	High	**0.970**	0.383	0.297	0.576	0.339	0.215	0.846
	4+4	RW	Low	**0.930**	0.610	0.503	0.561	0.505	0.319	0.870
	4+4	RW	High	**0.907**	0.437	0.349	0.542	0.334	0.217	0.899
	4+4	ER	Low	**0.970**	0.867	0.770	0.482	0.869	0.698	0.895
	4+4	ER	High	**0.970**	0.942	0.917	0.659	0.930	0.834	0.933
	4+6	AV	Low	**0.970**	0.389	0.277	0.354	0.284	0.217	0.822
	4+6	AV	High	**0.970**	0.257	0.223	0.577	0.225	0.197	0.817
	4+6	RW	Low	**0.930**	0.468	0.396	0.597	0.371	0.235	0.845
	4+6	RW	High	**0.907**	0.292	0.267	0.571	0.264	0.202	0.903
	4+6	ER	Low	**0.970**	0.860	0.756	0.494	0.810	0.645	0.882
	4+6	ER	High	**0.970**	0.937	0.896	0.621	0.907	0.773	0.931
Protein	5	——	——	**0.457**	0.452	0.441	**0.457**	0.439	0.424	0.441
Gene1	15	——	——	**0.703**	0.658	0.632	0.648	0.628	0.509	0.651
Gene2	15	——	——	0.838	0.83	0.809	0.734	0.460	0.229	**0.907**

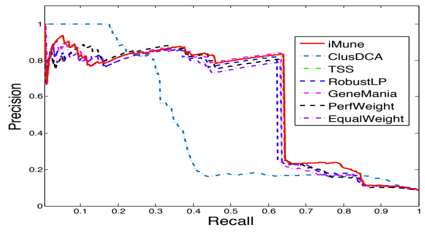
$c_u = 1$, and $c_- = 1 + sign(2p-1) * \gamma * \max(p, 1-p)$, where γ is a constant drawn from $[0.5, 1]$, and p is the proportion of class '$+1$' instances in the labeled set. For e.g., for $\gamma = 0.7$ and $p = 0.1$, we would have $c_+ = 1.63, c_u = 1, c_- = 0.37$.

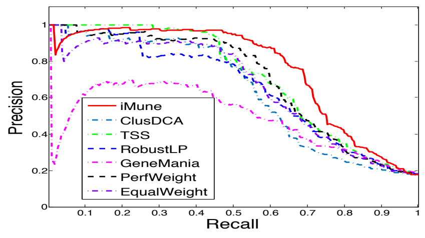
4.3 Evaluation Results

To perform semi-supervised classification, we label 5% of the nodes in *Protein*, *Gene1*, and *Gene2* and 30% in *RealityMining* which is a smaller dataset. We randomly sample the labeled set 10 times, and report the mean Average Precision (area under precision-recall curve) in Table 2 (notice in Table 1 that the datasets are class-imbalanced, hence accuracy is not a good measure to report). From the precision-recall plots in Fig. 2, we see that our method outperform baselines in almost all cases, which is especially evident in the presence of noise, when the performance of other methods degrade considerably. Interestingly, the baselines appear to be more robust against random noise than the other noise models.

We further investigate the effect of noise using *RealityMining* as a running example, as in the absence of noise all methods perform similarly on this multi-graph. Fig. 3 (left) shows how the performance of the methods change with increasing number of intrusive graphs (under rewiring and low-intensity). Fig. 3 (right) shows the same with


(a) *RealityMining*


(b) *Protein*


(c) *Gene1*

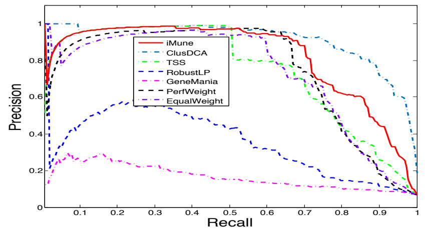
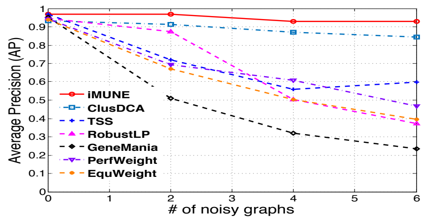

(d) *Gene2*

Fig. 2: *Noise hinders existing methods notably, whereas i*MUNE *remains near-stable.* Precision vs. Recall of competing methods in four real-world multi-graphs: (a) *RealityMining* (4 views), (b) *Protein* (5 views), (c) *Gene1* (15 views), and (d) *Gene2* (15 views). Inset plot in (a) shows performance when 6 rewired graphs with low intensity are injected to *RealityMining*



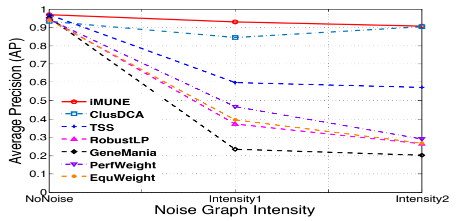


Fig. 3: *i*MUNE *performs better than all competitors* by (*left*) increasing number of intrusive graphs, and (*right*) increasing intensity of noise.

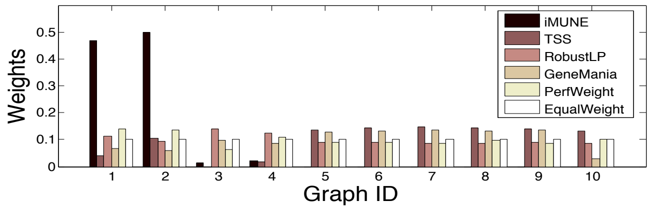


Fig. 4: *i*MUNE *filters out all noisy graphs, and gives large weights to most informative graphs. Competing methods are indifferent to noise and assign near-uniform weights.* Inferred graph weights on *RealityMining* (+6 injected noisy graphs G_5-G_{10} under AV and high-intensity).

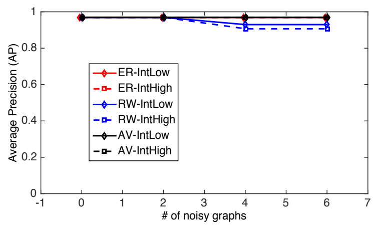


Fig. 5: iMUNE is robust under varying level (#graphs), intensity (low/high), models (ER,RW,AV) of noise.

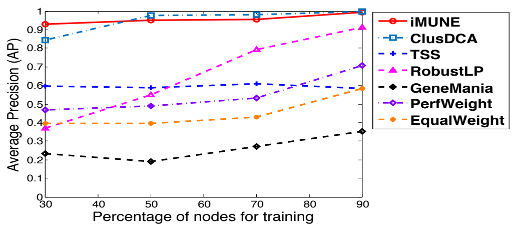


Fig. 6: Performance vs. % labeled nodes. iMUNE maintains high performance with different ratio of training data.

different noise intensity (under rewiring, 6 intrusive graphs). These show that iMUNE's performance remains near-stable, while the competing methods are relatively hindered by noise. In fact, as Fig. 5 shows iMUNE is robust under all settings: increasing level and intensity as well as different noise models.

We also analyze the inferred weights by each method (except *ClusDCA*, which adopts matrix factorization instead of learning graph weights). Fig. 4 shows the normalized weights on *RealityMining* with 6 injected graphs, under AV with high intensity.

Notice that all competing methods give non-zero weights to all the injected graphs G_5-G_{10}, which hinders their performance. In contrast, iMUNE puts non-zero weight only on the informative graphs G_1-G_4, particularly large weights on the first two. These are in fact the well-structured and denser informative graphs.

Finally in Fig. 6 we show how the performance of the methods change when we increase the labeled set percentage in *RealityMining* from 30% up to 90% (6 injected graphs, under rewiring with low intensity; results are avg'ed over 10 runs).

As expected the performance improves for all methods with increasing labeled data. However, most competing methods cannot achieve improved robustness and reach the same performance level by iMUNE, even when they are provided 90% of the data labeled. While *ClusDCA* achieves comparable performance when 50% of data is labeled, it is not as robust to noise as iMUNE as shown in Table 2.

5 Conclusion

In this work we introduced iMUNE, for robust, scalable, and effective *semi-supervised transductive classification for multi-relational graphs*. The proposed method employs a **convex formulation** that estimates weights for individual graphs, along with a solution that utilizes a weighted combination of them. Based on the **analysis of weights**, we devise a new scheme that iteratively discards intrusive graphs to achieve **robust** performance. Moreover, iMUNE is **linearly** scalable w.r.t. the size of the combined graph. Extensive experiments on real-world multi-graphs show that iMUNE produces competitive results under varying level, intensity, and models of noise. It also **outperforms** several baselines and state-of-the-art methods, which are notably hindered by the presence of noise where iMUNE remains immune.

Acknowledgments

This research is sponsored by NSF CAREER 1452425 and IIS 1408287. Any conclusions expressed in this material are of the authors and do not necessarily reflect the views, expressed or implied, of the funding parties.

References

1. Tsuda, K., Shin, H., Schölkopf, B.: Fast protein classification with multiple networks. Bioinformatics **21** (2005) 59–65
2. Kato, T., Kashima, H., Sugiyama, M.: Robust label propagation on multiple networks. IEEE Transactions on Neural Networks **20**(1) (2009) 35–44
3. Shin, H., Tsuda, K., Schölkopf, B.: Protein functional class prediction with a combined graph. Expert Syst. Appl. **36**(2) (2009) 3284–3292
4. Wan, M., Ouyang, Y., Kaplan, L., Han, J.: Graph regularized meta-path based transductive regression in heterogeneous information network. In: SDM, SIAM (2015)
5. Mostafavi, S., Ray, D., Warde-Farley, D., Grouios, C., Morris, Q.: GeneMANIA: a real-time multiple association network integration algorithm for predicting gene function. Genome Biology **9**(Suppl 1) (2008) S4
6. Mostafavi, S., Morris, Q.: Fast integration of heterogeneous data sources for predicting gene function with limited annotation. Bioinformatics **26**(14) (2010) 1759–1765
7. Luo, C., Guan, R., Wang, Z., Lin, C.: Hetpathmine: A novel transductive classification algorithm on heterogeneous information networks. In: Adv. in Info. Ret. (2014)
8. Lanckriet, G.R.G., Bie, T.D., Cristianini, N., Jordan, M.I., Noble, W.S.: A statistical framework for genomic data fusion. Bioinformatics **20**(16) (2004) 2626–2635
9. Argyriou, A., Herbster, M., Pontil, M.: Combining graph laplacians for semi-supervised learning. In: NIPS. (2005)
10. Yu, G.X., Rangwala, H., Domeniconi, C., Zhang, G., Zhang, Z.: Protein function prediction by integrating multiple kernels. In: IJCAI. (2013)
11. Wang, S., Jiang, S., Huang, Q., Tian, Q.: S3MKL: scalable semi-supervised multiple kernel learning for image data mining. In: ACM Multimedia, ACM (2010) 163–172
12. Barberá, P.: Birds of the same feather tweet together. bayesian ideal point estimation using twitter data. Political Analysis **23**(1) (2015) 76–91
13. Macskassy, S., Provost, F.: Classification in networked data: A toolkit and a univariate case study. Journal of Machine Learning Research **8** (2007) 935–983
14. Blum, A., Chawla, S.: Learning from labeled and unlabeled data using graph mincuts. In: ICML. (2001) 19–26
15. Zhu, X., Ghahramani, Z., Lafferty, J., et al.: Semi-supervised learning using gaussian fields and harmonic functions. In: ICML. (2003)
16. Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B.: Learning with local and global consistency. In: NIPS. (2003)
17. Belkin, M., Matveeva, I., Niyogi, P.: Regularization and semi-supervised learning on large graphs. In: COLT. (2004)
18. Spielman, D.A., Teng, S.H.: Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In: STOC, ACM (2004) 81–90
19. Eagle, N., Pentland, A.S., Lazer, D.: Inferring friendship network structure by using mobile phone data. PNAS **106**(36) (2009)
20. Wang, S., Cho, H., Zhai, C., Berger, B., Peng, J.: Exploiting ontology graph for predicting sparsely annotated gene function. Bioinformatics **31**(12) (2015) i357–i364