The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Overlap-Robust Decision Boundary Learning for Within-Network Classification

Sharad Nandanwar
Indian Institute of Science
Bangalore,India
sharadnandanwar@csa.iisc.ernet.in

M. N. Murty
Indian Institute of Science
Bangalore,India
mnm@csa.iisc.ernet.in

Abstract

We study the problem of within network classification, where given a partially labeled network, we infer the labels of the remaining nodes based on the link structure. Conventional loss functions penalize a node based on a function of its predicted label and target label. Such loss functions under-perform while learning on a network having overlapping classes. In relational setting, even though the ground truth is not known for the unlabeled nodes, some evidence is present in the form of labeling acquired by the nodes in their neighborhood. We propose a structural loss function for learning in networks based on the hypothesis that loss is induced when a node fails to acquire a label that is consistent with the labels of the majority of the nodes in its neighborhood. We further combine this with a novel semantic regularizer, which we call homophily regularizer, to capture the smooth transition of discriminatory power and behavior of semantically similar nodes. The proposed structural loss along with the regularizer permits relaxation labeling. Through extensive comparative study on different real-world datasets, we found that our method improves over the state-of-the-art approaches.

1 Introduction

The omnipresence of networks has led to a recent paradigm shift towards the analysis of network data. This has been witnessed by an influx of advancements for efficient storage and retrieval of such data. Presence of networks in day to day life is better observed through social networking sites (like Facebook, Twitter), e-Commerce sites (like Amazon, eBay), review sites (like Yelp, IMDb), etc. Network data models the relationship between the entities of a network, like the friendship between users, products co-purchased by customers, products reviewed by reviewers, etc. It is customary to represent network data in the form of a graph which is defined as follows:

Definition 1 *A network is modeled as a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of $|\mathcal{V}| = n$ interacting entities or nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges indicating relationship or interactions among the nodes.*

Modeling and analyzing the data in the form of networks has been shown to be effective across various domains; be it product recommendation (McAuley et al. 2015; Baluja et al. 2008), language models (Tang et al. 2015), web search (Page et al. 1999), patent analysis (Chakrabarti, Dom, and Indyk 1998), etc. However, analyzing these large complex networks is non-trivial and poses many challenges. Some of the key challenges involve social influence analysis, community detection in large graphs, inference of social ties, classification of entities in a network, etc. In this paper, we focus on the problem of within-network classification of entities. Entities in a network can be tagged based on their characteristic properties. These tags or labels are generally assigned manually. Nevertheless, it is very common to come across entities in a network which are not labeled. Inferring labels for such entities becomes crucial in many applications. Given a partially labeled network, the within-network classification problem is concerned with labeling unlabeled nodes. Formally it can be stated as

Definition 2 *Given a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ and a set of labeled nodes $\mathcal{V}_l (\subset \mathcal{V})$ with respective set of labels $\mathcal{Y}_l \in \Gamma^{|\mathcal{V}_l|}$ where $\Gamma = 2^{\{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k\}}$, for k classes labeled $\mathcal{C}_1, \ldots, \mathcal{C}_k$, the objective is to learn a model for inferring labels of unlabeled nodes $\mathcal{V}_u \subset \mathcal{V}$, where $\mathcal{V}_u = \mathcal{V} \setminus \mathcal{V}_l$.*

Classification of nodes in a relational setting is exemplified by graph-based semi-supervised learning methods (Macskassy, Provost, and Provost 2005), which infer label information of all the nodes collectively; collective classifiers are known to perform well. On the contrary, conventional classifiers like Support Vector Machine (SVM), which are known for their robustness, fail to deliver. These classifiers work on the assumption of independent and identically distributed patterns, which does not hold in a relational setting. In relational data, patterns to be classified are related to one another. Because of homophily in a network, it is expected that nodes related to each other will have similar behavior and hence the same class label.

Graph kernel based approaches (Kandola, Cristianini, and Shawe-taylor 2002; Smola and Kondor 2003; Kondor and Lafferty 2002) exploit homophily and capture network-based global similarity among nodes using a graph kernel. However, these methods do not scale well to deal with large networks. Nandanwar et al. proposed structural neighborhood based classification (Nandanwar and Murty 2016), which performs recursive classification using a global neighborhood of the node under consideration. However, the ap-

proach fails on networks with overlapping classes. In these cases although only few labels are observed for majority of the nodes, they can actually have significant soft membership in many more classes. Conventional hinge loss infers a zero loss if the predicted label matches with the observed label. Due to this unobserved soft membership, when a classifier makes correct prediction for a node, it simultaneously also makes an error by not predicting other classes. In this work, we define a structural loss function and its efficient minimization for learning the optimal decision boundary in networks with overlapping classes. The proposed structural loss function is based on the principle of homophily and ensures that the neighborhood of a node exhibits the same behavior as the node. The main contributions of our work are:

- *We propose for the first time a structural loss function which exploits the network structure for penalizing misclassification of nodes in a novel manner, while simultaneously offering a practical treatment of boundary nodes.*
- *We provide an efficient algorithm for minimizing the proposed structural loss combined with a novel homophily regularizer.*
- *We carry out an extensive comparative study and show the effectiveness of the proposed approach.*

The rest of this paper is organized as follows. In Section 2 we briefly survey current state-of-the-art. We define structural loss for classification in Section 3, and propose an algorithm for the same. Section 4 describes the experimental setup. We present our empirical findings in Section 5, followed by conclusion in Section 6. We make our code and datasets publicly available at `https://github.com/sharadnandanwar/ReSLMin`.

2 Related Work

Classification of linked entities in a network has been an active area of research in machine learning since long. The problem is often referred to as collective classification because of the involved transductive setting, where labels of all the unlabeled nodes are determined simultaneously. Collective classification leads to a significant reduction in classification error (Jensen, Neville, and Gallagher 2004). Our problem falls under the broad purview of semi-supervised learning. One of the seminal works on graph based semi-supervised learning includes (Zhou et al. 2004).

Classification of scientific patents was studied by Chakrabarti et al. (Chakrabarti, Dom, and Indyk 1998), where local features of documents were augmented using features from small-radii neighborhoods. Macskassy and Provost(Macskassy, Provost, and Provost 2005) described Network-Only-Bayes classifier similar to the above but without using local features. Collective classification approaches can be broadly categorized into iterative classification and relaxation labeling. Lu and Getoor (Lu and Getoor 2003) proposed an iterative algorithm for link based classification where features based on link and object are computed at each iteration and a new labeling is inferred using conventional learning approaches like logistic regression. In (Macskassy and Provost 2003) weighted-vote relational neighbor (wvRN) classifier was introduced which defines the class membership as a weighted average of the class membership probabilities of the neighboring nodes. Another relational classifier Multi Rank Walk (Lin and Cohen 2010) adopts a random walk perspective identical to the Page Rank algorithm. (Zhou et al. 2004) borrows the idea from spectral clustering that linked nodes should have a smooth transitioning. It proposes an iterative method for updating label information of a node using information from neighboring nodes and exploits further a regularization framework for the same. Label propagation algorithm (Zhu and Ghahramani 2002) defines a probabilistic process for labels to transit between nodes with a combination of random walk and clamping. For discovering and suggesting videos, Baluja et al. proposed Adsorption algorithm (Baluja et al. 2008) which takes a controlled random walk over the graph. Talukdar et al. observed that Adsorption algorithm does not guarantee convergence and proposed a modification to it (Talukdar and Crammer 2009) by defining a well-behaved objective function for minimization.

Collective classification strongly relies on the homophily assumption. In (Sen et al. 2008) the authors argued that in a sparsely labeled network relational summary of a node may be meaningless and proposed a collective prediction framework by exploiting latent linkages in a network. According to cluster hypothesis in a network, nodes in the same community are likely to be of the same class. SocioDim (Tang and Liu 2009; 2011) attempts to learn from social dimensions which capture hidden semantics discovered using linkage structure of the network. However, these approaches fail on networks where cross linkages are significantly high. In (Wang and Sukthankar 2013) a probabilistic method similar to weighted vote relational neighbor classifier is proposed which exploits the social context features while computing posterior probability for each node. A generalized framework for unsupervised feature learning, DeepWalk (Perozzi, Al-Rfou, and Skiena 2014), computes social representation by taking short random walks. Related to this, LINE(Tang et al. 2015) suggests an efficient scalable alternative for computing low dimensional network embedding using second order proximity in the network. These embeddings capturing the neighborhood and cluster property are further used in node classification. More recently node2vec (Grover and Leskovec 2016) was proposed which preserves the neighborhood of nodes in a low-dimensional embedding.

Another class of methods which advocate homophily assumption are graph kernel based approaches. A graph kernel captures the similarity between nodes in a network. Graph kernels like Von-Neumann and exponential graph kernels (Kandola, Cristianini, and Shawe-taylor 2002) find similarity between nodes by aggregating the number of paths between two nodes while giving lesser importance to longer paths. Laplacian counterparts of the above were introduced on similar lines (Smola and Kondor 2003; Kondor and Lafferty 2002). A similar discriminative random walk based approach which relies on betweenness w.r.t. a class was described in (Callut et al. 2008). The disadvantage, however, is that graph kernel based approaches do not scale well to handle large networks.

3 Regularized Structural Loss Minimization

Conventional learning approaches based on Support Vector Machine, Logistic Regression, k-Nearest Neighbors, etc., when applied to a network setting, use local adjacency information of nodes as features. While doing so, these approaches emphasize that if node v_i having label y is related to node v_j, and node v_k is related to v_j then v_i and v_k should have the same class label y. While this sounds intuitive and in accordance with common neighbor similarity, still it fails to emphasize on facts like v_j should have the same class label as v_i and v_k. This approach works well if the number of labeled nodes in the network is sufficiently large. However, when a network is sparsely labeled, this fails. This is because, the pairs of labeled nodes having common neighbors become increasingly rare with increase in sparsity.

We introduce the notation and conventions employed in this paper now. Consider an undirected binary weighted graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$. The adjacency matrix $\mathbf{A}$ corresponding to $\mathcal{G}$ is defined as,

$$A_{ij} = \begin{cases} 1 & \text{if } (v_i, v_j) \in \mathcal{E} \\ 0 & \text{otherwise} \end{cases}.$$

Column $\mathbf{a}_i$ of $\mathbf{A}$ represents the adjacency vector of node v_i. Further, let $\mathcal{N}_i$ represent the immediate (first-level) neighborhood of node v_i i.e. $\mathcal{N}_i = \{v_j : a_{ij} = 1\}$. We define a more generalized k^{th}-level neighborhood of a node as follows

Definition 3 *The* $\boldsymbol{k^{th}}$***-Level Neighborhood*** *of a node v_p is defined as a multiset $\mathcal{N}_p^k$ s.t. $v_r \in \mathcal{N}_p^k$ if and only if there is an edge in graph $\mathcal{G}$ connecting nodes v_r and v_q, where node $v_q \in \mathcal{N}_p^{k-1}$*

Throughout this section, we would be working with the binary classification problem. Let $\mathcal{C}_+$ and $\mathcal{C}_-$ represent the positive and negative classes. In our experiments we deal with multi-class classification, which can be handled by a combination of binary classifiers using *"one-vs-one"* or *"one-vs-rest"* approach (Bishop 2006).

3.1 Structural Loss

For a collection of patterns $\{x_1, x_2 \ldots, x_N\}$ with labels $\{y_1, y_2, \ldots, y_N\}$ respectively, where, $x_i \in \mathbb{R}^n$ and $y_i \in \{+1, -1\}$, the objective of classification is to learn a function $f : \mathbb{R}^n \to \mathbb{R}$ which minimizes the empirical risk given by

$$\mathcal{R} = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(f(x_i), y_i).$$

Depending on the application and geometry of data, many loss functions have been introduced in the past. Some of the popular ones include Hinge Loss, Squared Loss, and Logistic Loss. Earlier, many studies have established the robust behavior of the hinge loss function compared to others. Henceforth, while explaining the proposed structural loss function, we will use a function similar to the hinge loss.

The hinge loss function for a node v_i for classification based on its adjacency representation a_i is given as follows

$$\xi_i = \max(0, 1 - f(a_i) y_i).$$

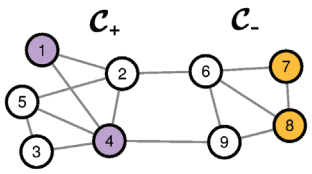


Figure 1: A labeled toy network for binary classificaiton problem.

Since we would be interested in a linear decision boundary, the function f can be written as $f(a_i) = w^\mathsf{T} a_i + b$, where w is the weight vector and b is the threshold. Thus,

$$\xi_i = \max(0, 1 - y_i(w^\mathsf{T} a_i + b)). \tag{1}$$

Let us denote this self-induced loss (i.e. loss induced by its own misclassification) for node v_i by L_i^0. Then, we have $L_i^0 = \xi_i$.

In accordance with the principle of homophily, the neighboring nodes of a node are expected to have the same class label. A loss is induced if any of the neighbors of a node fail to have the same label as the node. Let ξ_{ij} be the loss when node v_j fails to acquire the same class label as its neighbor v_i. We define ξ_{ij} as follows

$$\xi_{ij} = \max(0, 1 - y_i(w^\mathsf{T} a_j + b)),$$

Based on this, L_i^0 can be equivalently written as $L_i^0 = \xi_i = \xi_{ii}$. The expected loss (L_i^1) induced by labeled node v_i in its first level neighborhood is given by,

$$L_i^1 = \frac{1}{|\mathcal{N}_i|} \sum_{v_j \in \mathcal{N}_i} \xi_{ij}$$

Similarly, as we move further away from the labeled node under consideration, for second level neighborhood of v_i, the expected induced loss (L_i^2) is given by

$$L_i^2 = \frac{1}{|\mathcal{N}_i|} \sum_{v_{j_1} \in \mathcal{N}_i} \frac{1}{|\mathcal{N}_{j_1}|} \sum_{v_{j_2} \in \mathcal{N}_{j_1}} \xi_{ij_2}$$

Generalizing this to k^{th}-level neighborhood we have

$$L_i^k = \frac{1}{|\mathcal{N}_i|} \sum_{v_{j_1} \in \mathcal{N}_i} \frac{1}{|\mathcal{N}_{j_1}|} \sum_{v_{j_2} \in \mathcal{N}_{j_1}} \cdots \frac{1}{|\mathcal{N}_{j_{k-1}}|} \sum_{v_{j_k} \in \mathcal{N}_{j_{k-1}}} \xi_{ij_k}$$

For further clarity, we illustrate the notation using a toy network shown in Figure 1. While classifying nodes in the above network, a loss is incurred if a node belonging to $\mathcal{C}_+$ is predicted to be in $\mathcal{C}_-$ or vice versa. For instance, consider the labeled node **1**, hinge loss corresponding to which is given by ξ_1, where ξ_1 is defined by equation 1. As explained above, in a network setting the labeled node **1** induces a loss for its neighbors also. For first level neighbors, the loss is given by $L_1^1 = \dfrac{\xi_{1,2} + \xi_{1,4}}{2}$. Similarly we have $L_1^2 = \frac{1}{2}\left(\frac{\xi_{1,1} + \xi_{1,4} + \xi_{1,5} + \xi_{1,6}}{4} + \frac{\xi_{1,1} + \xi_{1,2} + \xi_{1,3} + \xi_{1,5} + \xi_{1,9}}{5}\right)$, for second level neighbors and so on.

Aggregating all these losses using $\alpha \in [0, 1]$ for weighing the neighborhood effect, the final observed empirical risk is

$$\mathcal{R} = \frac{1}{|\mathcal{V}_l|} \sum_{i \in \mathcal{V}_l} \left(\alpha^0 L_i^0 + \alpha^1 L_i^1 + \alpha^2 L_i^2 + \ldots\right).$$

For a given undirected graph, by rearranging the terms the above can be rewritten as

$$\mathcal{R} = \frac{1}{|\mathcal{V}_l|}\sum_{v_i\in\mathcal{V}_l}\Bigg(\alpha^0\xi_{ii} + \alpha^1\sum_{i_1\in\mathcal{N}_i\cap\mathcal{V}_l}\frac{\xi_{i_1i}}{|\mathcal{N}_{i_1}|} + \alpha^2\sum_{i_1\in\mathcal{N}_i}\frac{1}{|\mathcal{N}_{i_1}|}\sum_{i_2\in\mathcal{N}_{i_1}\cap\mathcal{V}_l}\frac{\xi_{i_2i}}{|\mathcal{N}_{i_2}|} + \cdots + \alpha^k\sum_{i_1\in\mathcal{N}_i}\frac{1}{|\mathcal{N}_{i_1}|}\cdots\sum_{i_k\in\mathcal{N}_{i_{k-1}}\cap\mathcal{V}_l}\frac{\xi_{i_ki}}{|\mathcal{N}_{i_k}|} + \cdots\Bigg)$$

The above can also be seen as reverse cascading of loss in a network. Ignoring the constant part $\frac{1}{|\mathcal{V}_l|}$, we call the term in the outer summation of the above equation as the structural loss for node v_i and denote it by σ_i. The set of labeled nodes $\mathcal{V}_l$ can be partitioned into two sets based on $\mathcal{C}_+$ and $\mathcal{C}_-$. Let us denote the loss induced on node v_i by positively labeled nodes ($\subset\mathcal{C}_+$) as σ_i^+ and loss induced by negatively labeled nodes ($\subset\mathcal{C}_-$) as σ_i^-, s.t. $\sigma_i = \sigma_i^+ + \sigma_i^-$. Then

$$\sigma_i^+ = \Bigg(\alpha^0\xi_{ii} + \alpha^1\sum_{i_1\in\mathcal{N}_i\cap\mathcal{V}_l\cap\mathcal{C}_+}\frac{\xi_{i_1i}}{|\mathcal{N}_{i_1}|} + \alpha^2\sum_{i_1\in\mathcal{N}_i}\frac{1}{|\mathcal{N}_{i_1}|}\sum_{i_2\in\mathcal{N}_{i_1}\cap\mathcal{V}_l\cap\mathcal{C}_+}\frac{\xi_{i_2i}}{|\mathcal{N}_{i_2}|} + \cdots + \alpha^k\sum_{i_1\in\mathcal{N}_i}\frac{1}{|\mathcal{N}_{i_1}|}\cdots\sum_{i_k\in\mathcal{N}_{i_{k-1}}\cap\mathcal{V}_l\cap\mathcal{C}_+}\frac{\xi_{i_ki}}{|\mathcal{N}_{i_k}|} + \cdots\Bigg).$$

Let $\vec{c^+}$ be the vector defined as $c_i^+ = \begin{cases}1 & \text{if } v_i\in\mathcal{V}_l\cap\mathcal{C}_+\\ 0 & \text{otherwise}\end{cases}$.

Then vector $\vec{\sigma^+}$ corresponding to the above can be written using matrix algebra as

$$\vec{\sigma^+} = \vec{\beta^+}\bigodot\max\left(\vec{0},\vec{1}-(A\vec{w}+b\vec{1})\right),$$

where $\bigodot$ represents element-wise (Hadamard) product operator for vectors and

$$\vec{\beta^+} = (\alpha^0 I + \alpha^1 AD^{-1} + \alpha^2(AD^{-1})^2 + \dots)\vec{c^+}.$$

where D is a diagonal matrix given by

$$D = diag(|\mathcal{N}_1|,\ |\mathcal{N}_2|,\ \dots,\ |\mathcal{N}_{|\mathcal{V}|}|).$$

Similarly, $\vec{c^-}$ is defined as $c_i^- = \begin{cases}1 & \text{if } v_i\in\mathcal{V}_l\cap\mathcal{C}_-\\ 0 & \text{otherwise}\end{cases}$,

and vector $\vec{\sigma^-}$ will be

$$\vec{\sigma^-} = \vec{\beta^-}\bigodot\max\left(\vec{0},\vec{1}+(A\vec{w}+b\vec{1})\right)$$

where

$$\vec{\beta^-} = (\alpha^0 I + \alpha^1 AD^{-1} + \alpha^2(AD^{-1})^2 + \dots)\vec{c^-}$$

Thus structural loss for each node is given by,

$$\begin{aligned}\sigma_i &= \sigma_i^+ + \sigma_i^-\\ &= \beta_i^+\max\left(0,1-(\vec{a_i}^\mathsf{T}\vec{w}+b)\right) + \beta_i^-\max\left(0,1+(\vec{a_i}^\mathsf{T}\vec{w}+b)\right)\end{aligned} \quad (2)$$

A node with higher degree will have larger number of labeled neighbors as well as larger number of short paths to labeled nodes. So, it is easy to observe that for such high degree nodes, value of β_i^+ or β_i^- will be much higher compared to those of low degree nodes. To remove this bias we normalize β_i's for all nodes so that $\beta_i^+ + \beta_i^- = 1$. Figure 2 illustrates the proposed structural loss function graphically.

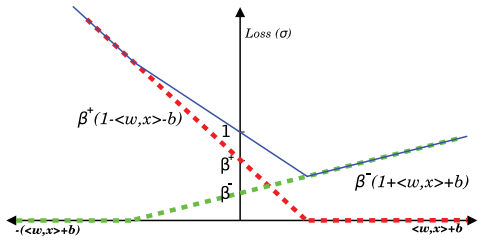


Figure 2: Illustration of Structural Loss Function. The dotted red (green) line represents the loss function σ^+ (σ^-), corresponding to $+^{\text{ve}}$ ($-^{\text{ve}}$) class. And, the solid blue line is the overall loss (σ).

3.2 Homophily Regularization

Minimizing empirical risk alone overfits the model and does not generalize well. Regularization is a popular technique used for controlling the model complexity. The most commonly used regularizer is ℓ_2 (ridge) regularizer, which also corresponds to maximum margin regularization. However, in a relational setting regularizers like laplacian regularizer (Smola and Kondor 2003) have been found to perform well. Laplacian regularizer emphasizes on the smoothness of predicted function value by penalizing sudden jumps in the function values of connected nodes.

In relational learning, the small number of labeled examples combined with sparse neighborhood of nodes may prohibit the learning of actual discriminative behavior of nodes. To counter this effect we use a regularizer and coin it as *homophily regularizer*. The regularizer minimizes

$$\sum_{(v_i,v_j)\in\mathcal{E}}(w_i-w_j)^2 = w^\mathsf{T}Lw$$

L is the graph laplacian matrix given by $L = D - A$, where D is the degree matrix and A is the adjacency matrix of the graph. The homophily regularizer term reinforces a smooth transition in the discrimination power (captured by w) of connected nodes. In addition, it is also desired to attenuate the effect of noisy links to outlier nodes. Considering this we retain the conventional ℓ_2 regularizer, which helps in diluting the weights assigned to such outlier nodes.

3.3 Learning Objective

The objective for the proposed structural loss function along with homophily regularizer is given by,

$$\min_{\vec{w},b}\quad\frac{\lambda}{2}||w||^2 + \frac{\mu}{2}w^\mathsf{T}Lw + \frac{1}{|\mathcal{V}_l|}\sum_{v_i\in\mathcal{V}}\sigma_i$$

where σ_i is defined by equation 2.

Note that the proposed objective function is convex as it is expressed as a non-negative sum of convex functions. However, the loss function is not smooth. Using gradient descent method the update rules for $\vec{w}$ and b are derived as

$$\begin{aligned}\vec{w}_{t+1} &= \vec{w}_t - \eta_t\left(\lambda\vec{w}_t + \mu L\vec{w}_t + \frac{1}{|\mathcal{V}_l|}\sum_{v_i\in\mathcal{V}}[\nabla_{\vec{w}}\sigma_i]_t\right)\\ b_{t+1} &= b_t - \frac{\eta_t}{|\mathcal{V}_l|}\sum_{v_i\in\mathcal{V}}[\nabla_b\sigma_i]_t\end{aligned}$$

Algorithm 1 Training ReSLMin using SGD.

Given: Adjacency matrix $\mathbf{A}$, Training set Tr, Label vector y, Learning parameters λ, μ.

$\mathbf{D} \leftarrow diag(\overrightarrow{degree})$

$\vec{c^+} \leftarrow (c_1^+, c_2^+, \ldots, c_n^+)$ where, $c_i^+ = \begin{cases} 1 & \text{if } y_i = 1, \\ 0 & \text{otherwise.} \end{cases}$

$\vec{c^-} \leftarrow (c_1^-, c_2^-, \ldots, c_n^-)$ where, $c_i^- = \begin{cases} 1 & \text{if } y_i = -1, \\ 0 & \text{otherwise.} \end{cases}$

$\vec{\beta}^+, \vec{\beta}^- \leftarrow \vec{0}_{n\times 1}$

while change$(\vec{\beta^+}) > \epsilon$ **or** change$(\vec{\beta^-}) > \epsilon$ **do**

$\quad \vec{\beta}^+ \leftarrow \vec{\beta}^+ + \alpha^i(\mathbf{AD}^{-1})^i\vec{c}^+$

$\quad \vec{\beta}^- \leftarrow \vec{\beta}^- + \alpha^i(\mathbf{AD}^{-1})^i\vec{c}^-$

$\vec{w}_0 \leftarrow \vec{0}_{n\times 1}, b \leftarrow 0$

while not converged **do**

Randomly choose $Smp \subseteq Tr$, of size k

$Mc^+ \leftarrow \{i \in Smp : (\vec{x}_i^\mathsf{T}\vec{w}_{t-1} + b) < 1\}$

$Mc^- \leftarrow \{i \in Smp : (\vec{x}_i^\mathsf{T}\vec{w}_{t-1} + b) > -1\}$

$$\vec{w}_t \leftarrow (I - \eta_t\lambda I - \eta_t\mu L)\vec{w}_{t-1} + \frac{\eta_t}{|Tr|}\cdot\frac{n}{k}\left(\sum_{i\in Mc^-}\beta_i^-\vec{x}_i - \sum_{i\in Mc^+}\beta_i^+\vec{x}_i\right)$$

$$b_t \leftarrow b_{t-1} + \frac{\eta_t}{|Tr|}\cdot\frac{n}{k}\left(\sum_{i\in Mc^-}\beta_i^- - \sum_{i\in Mc^+}\beta_i^+\right)$$

return $\vec{w}, b$ (w and b are entities after convergence)

where $[\nabla_{\vec{w}}\sigma_i]_t$ and $[\nabla_b\sigma_i]_t$ denote the sub-gradient of structural loss for node v_i w.r.t $\vec{w}$ and b respectively at t^{th} iteration, and is computed as,

$$\nabla_{\vec{w}}\sigma_i = \begin{cases} \beta_i^-\vec{a}_i & \text{if } \vec{a}_i^\mathsf{T}\vec{w} + b > 1 \\ (\beta_i^- - \beta_i^+)\vec{a}_i & \text{if } -1 < \vec{a}_i^\mathsf{T}\vec{w} + b < 1 \\ -\beta_i^+\vec{a}_i & \text{if } \vec{a}_i^\mathsf{T}\vec{w} + b < -1 \end{cases}$$

$$\nabla_b\sigma_i = \begin{cases} \beta_i^- & \text{if } \vec{a}_i^\mathsf{T}\vec{w} + b > 1 \\ (\beta_i^- - \beta_i^+) & \text{if } -1 < \vec{a}_i^\mathsf{T}\vec{w} + b < 1 \\ -\beta_i^+ & \text{if } \vec{a}_i^\mathsf{T}\vec{w} + b < -1 \end{cases}$$

3.4 Algorithm and Complexity

Algorithm 1 summarizes the pseudo-code for training the proposed model **ReSLMin**. We next analyze the run time complexity of the proposed algorithm. For obtaining an ϵ-accurate solution to a strongly convex objective, the number of iterations required by stochastic gradient descent is $O(\frac{1}{\epsilon})$. In *ReSLMin*, prior generation of relaxed labels requires $O(nd)$ computation at each level of neighborhood, where n is the number of nodes and $d(<< n)$ is average degree of nodes in the given network. For a batch of size k, each epoch requires $O(kn)$ computation in the case of ridge regularizer and $O(kn+nd)$ in case of homophily regularizer. The overall complexity of proposed algorithms is $O(\frac{nd}{\epsilon})$. In Figure 3 depicts the behavior of generalization error, measured as $(1 -$ Hamming Score$)$ at each iteration.

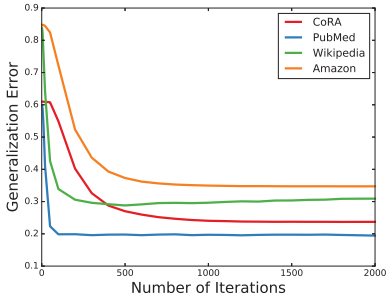


Figure 3: Convergence behavior of ReSLMin Algorithm, shown by plotting **(1-Hamming Score)** vs. **number of SGD iterations**.

Dataset	#Nodes	#Edges	#Classes	Label Cardinality
PubMed	19,717	44,324	3	1.000
CoRA	24,519	92,207	10	1.004
Wikipedia	35,633	495,388	16	1.312
Amazon	83,742	190,097	30	1.546

Table 1: Datasets used in Experiments.

4 Experiments

We evaluate the impact of using the proposed structural loss function using different regularizers by observing its performance on various real-world datasets. To make our experimental study exhaustive, we compare with various state-of-the-art techniques for within network classification.

4.1 Data Sets

We next describe the datasets used in our experiments.

Co-citation Graph: We take two different instances of academic co-citation graph viz. CoRA and PubMed (Sen et al. 2008). Classes are assigned based on the subtopic or field-of-study the papers belong to. In both the datasets, edges correspond to the references made by the papers.

Product Graph: We extracted a subset of books (with > 5 reviews) from Amazon co-purchasing network data (Leskovec, Adamic, and Huberman 2007). For each book, we define the edges using list of similar books. We use genres of the book as class labels.

Web Graph: We constructed a web-graph of Wikipedia pages. Articles from different areas of computer science were crawled using WikiMedia API. While crawling we used 16 top level category pages, which also form the class labels.

For networks having directional edges, we remove the directionality by adding an edge in the opposite direction also. Table 1 summarizes some of the statistics of datasets used.

4.2 Approaches

We make a comparative study of *ReSLMin*, combined with different regularizers. Further, we also evaluate the performance of our proposed approach with the state-of-the-art techniques mentioned in (Nandanwar and Murty 2016). We briefly mention these approaches below.

Dataset	Measure	ReSLMin				SDEC	SCRN	DW	SNBC
		Lap.	Ridge	Homo.	Ridge + Homo.				
PubMed	Hamming	75.58±0.09	80.23±0.10	80.58±0.04	**80.83±0.04**	56.89±0.01	79.56±0.14	78.06±0.09	80.04±0.19
	Micro F_1	75.58±0.09	80.23±0.10	80.58±0.04	**80.83±0.04**	56.89±0.01	79.56±0.14	78.06±0.09	80.04±0.19
	Macro F_1	74.10±0.13	78.55±0.13	78.91±0.07	**79.33±0.05**	42.25±0.01	77.99±0.19	76.21±0.10	78.28±0.24
CoRA	Hamming	74.67±0.09	76.45±0.07	75.25±0.14	**77.52±0.07**	55.32±0.07	74.93±0.17	71.82±0.20	67.39±2.23
	Micro F_1	74.67±0.09	76.43±0.07	75.24±0.15	**77.49±0.07**	55.36±0.07	74.93±0.17	71.82±0.20	67.39±2.24
	Macro F_1	66.30±0.29	70.05±0.14	66.52±0.22	**71.03±0.24**	33.14±3.61	66.16±0.35	62.93±0.51	57.47±3.96
Wikipedia	Hamming	67.45±0.10	69.94±0.14	71.06±0.25	**72.25±0.11**	53.04±0.08	70.40±1.82	72.12±0.08	69.92±0.16
	Micro F_1	69.19±0.08	71.67±0.12	72.38±0.32	**73.65±0.11**	55.27±0.05	71.98±1.22	73.65±0.07	71.32±0.12
	Macro F_1	61.06±0.54	64.16±0.44	65.81±0.65	**66.58±0.53**	39.49±0.08	63.59±1.24	64.99±0.65	61.38±1.58
Amazon	Hamming	66.19±0.06	66.44±0.08	**66.95±0.05**	66.49±0.05	35.85±0.06	65.31±0.07	29.77±0.03	61.24±0.19
	Micro F_1	66.56±0.04	66.77±0.06	**67.03±0.04**	66.75±0.03	37.58±0.06	66.51±0.05	31.98±0.03	61.71±0.25
	Macro F_1	63.07±0.13	63.46±0.12	63.72±0.08	**63.86±0.05**	23.62±0.21	63.14±0.06	20.25±0.10	59.69±0.31

Table 2: Performance evaluation of various approaches. The rightmost four approaches correspond to the different regularization techniques used with the structural loss. The remaining four on the left, are state-of-the-art approaches chosen for comparison.

DeepWalk (*DW*) (Perozzi, Al-Rfou, and Skiena 2014): We use Deepwalk for learning a deep representation of nodes in the network. Empirically we found 128-dimensional representations to have the best behavior for our classification problem. These representations cater to the input for multi-class support vector machine.

SocioDim Edge Clustering (*SDEC*) (Tang and Liu 2011): *SDEC* generates a k- dimensional representation based on the clustering of edges in the network. We empirically determined the best value of k as 5000 for all datasets.

Social Context Relational Neighbor (*SCRN*) (Wang and Sukthankar 2013): *SCRN* computes low dimensional embedding by combining ideas from wvRN classifier and edge clustering. We empirically fix the number of edge clusters as 5000.

Structural Neighborhood Based Classification (*SNBC*) (Nandanwar and Murty 2016): We use the *SNBC* source code available from authors repository. We tune the parameters using grid search to learn a model for multilabel classification.

4.3 Setup

The datasets used in our experiments have a set of labels assigned to each node. We handle multilabel classification problem by training a series of binary classifiers using *"one-vs-rest"* approach. The scores returned by functional classifier are turned into probabilities using *"Platt scaling"*. This is required to enable comparison of scores from different classifiers for a given node.

The hypothesis of our approach is that a loss should be induced from a neighboring node when the neighbor node fails to acquire a similar label. In Section 3 we work with a uniformly induced loss i.e. an equal weight is given to loss induced by any neighboring node. However, it is well established that connections to low degree nodes are semantically rich compared to their high degree counterparts. Considering this we assign non-uniform weights to loss from different neighboring nodes. A node with degree d induces a loss inversely proportional to $\log(d)$, thus underplaying the contribution of loss induced from high degree neighbors.

Further, for the proposed structural loss function, we experiment with different choices of regularizers, namely, laplacian, ridge, and the proposed homophily regularizer along with its combination with ridge regularizer. To determine the optimal values of regularization parameter, we perform 10-fold cross-validation with the given set of training nodes. The parameter values are chosen using grid search technique where λ and μ both were in the range $\{2^{-5}, 2^{-6}, \ldots, 2^{-15}\}$. Also, we set the value of parameter α, responsible for neighborhood effect, as 0.5. For other state-of-the-art approaches, to make a fair comparison, we tune parameters in a similar manner.

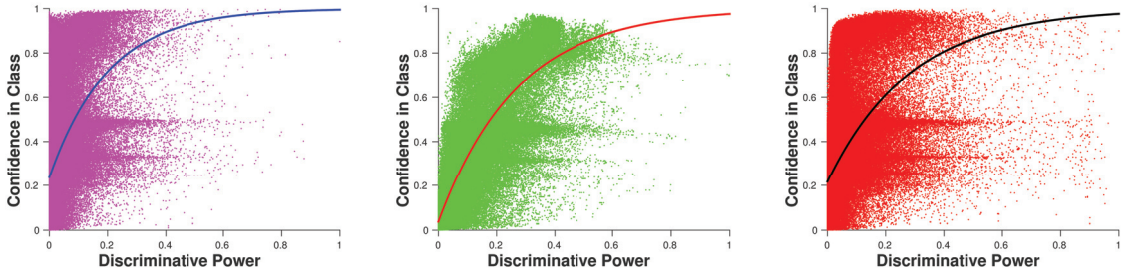
In our multilabel setting, each node has a different number of labels assigned to it. For an unlabeled node, we predict a set of labels using predicted class membership probabilities. The cardinality of the predicted label set is equal to that of its original label set. For evaluation of multilabel classification, we use three evaluation measures namely micro-F_1 score, macroF_1 score, and hamming score (Sokolova and Lapalme 2009).

For each experiment, we perform 20 runs using different realizations of the train and test sets, having 10% and 90% of nodes respectively. We report the mean and deviation of observed scores in Table 2.

5 Results

We first analyze the behavior of various state-of-the-art techniques compared against the proposed *ReSLMin* algorithm. From Table 2, it is easy to follow that *ReSLMin* combined with *ridge+homophily* regularizer consistently outperforms all other approaches for most of the datasets considered in our experiments. This corroborates our homophily assumption which controls both loss function and regularization.

We find that socioDim edge clustering (*SDEC*) loses in almost every case. This hints that the semantics captured by *SDEC* in spherical edge clusters are not sufficient for classification. However, *SCRN* which incorporates same edge cluster information with collective classification tends to



a) Ridge b) Homophily c) Ridge + Homophily

Figure 4: For Amazon dataset the plots highlight the role played by homophily regularizer. The scatter plots depict for all nodes **confidence in class** vs. **discriminative power**. The middle plot corresponding to homophily regularizer corroborates that the regularizer helps in achieving the proposed hypothesis i.e. *Nodes having high confidence value should have high discriminative power*. The curve in plots indicates the least square fit of the form $(1 - ae^{bx})$. The other datasets also follow a similar trend (omitted due to space constraint).

perform better.

The representations captured by *DeepWalk* perform comparatively better but fail drastically on `Amazon` dataset. Our primary analysis reveals that the deepwalk embeddings fails to capture the community structure because of high overlap across genres of books.

Structural neighborhood based classification (*SNBC*) performs poorly on all datasets but `PubMed` (which has least class overlap). This is due to improper penalizing of boundary nodes which we address in *ReSLMin*.

5.1 Comparative Study of Regularizers

We next study the effect of adding different regularizers to structural loss. In all but `Amazon` dataset laplacian regularizer lags by a heavy margin. Laplacian regularizer fails to generalize well for unseen nodes as it directly tries to smoothen the decision values, encouraging overfitting. While, in the case of homophily regularizer, smoothening is done on parameters determining the discriminative behavior. Evaluation metrics reported in Table 2 clearly indicate the superiority of combined regularizer over the individual counterparts and laplacian regularizer. Surprisingly, in case of `Amazon` dataset homophily regularizer does better than the combination. This can be attributed to the discriminative power getting highlighted by homophily regularizer.

We carry on this analysis, and plot the discriminative power of node vs its confidence probability of being in a class. For a node v_i, we use the magnitude of its weight ($|w_i|$) as a measure of *discriminative power*. Since the ground-truth about class membership probabilities of nodes is not available to us, we use the predicted class membership probabilities as a proxy to the *confidence in class*.

Figure 4 shows the plots corresponding to considered regularization techniques for `Amazon` dataset. It is easy to observe from the plots that ridge (ℓ_2) regularizer assigns lesser importance to most of the points, even in cases where nodes have high class membership values. On the other hand homophily regularizer emphasizing smooth transition assigns a higher importance to nodes with high class membership values. The combination of the two tries to balance a trade-off.

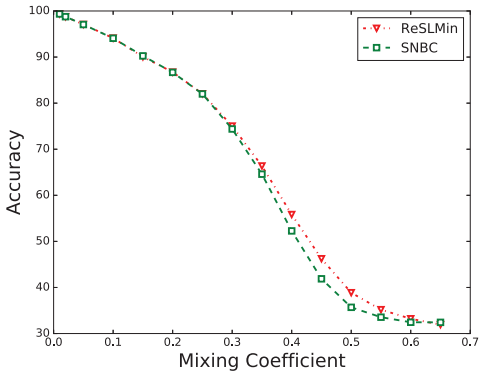


Figure 5: Comparison of SNBC and ReSLMin Algorithms as mixing coefficient is varied in LFR Benchmark Synthetic data.

The combined regularizer allows a smooth transition among nodes lying at the core of the network, while at the same time assigning little or no importance to peripheral nodes. The importance of peripheral nodes is shadowed by ridge regularizer because of their insignificant role.

5.2 Robustness of ReSLMin

For studying the robustness of the proposed approach with increase in class overlap, we synthesize a network of 10000 nodes using LFR benchmark (Lancichinetti, Fortunato, and Radicchi 2008). We vary the degree of overlap in the network by changing the value of mixing coefficient parameter $\mu \in [0, 1]$. Value $\mu = 0$ corresponds to k connected components in the network, while $\mu = 1$ corresponds to k-partite graph. Similar to the earlier described setting we randomly choose 10% of nodes for training, and infer labeling of the remaining 90% nodes. We make a comparative study with SNBC, which is very similar in spirit to the proposed approach. It can be noticed from Figure 5 that both these algorithms achieve the same performance when overlap is low. As the mixing coefficient is increased beyond 0.25, the su-

periority of ReSLMin becomes evident. However, on further increasing the mixing coefficient value beyond 0.65 both algorithms fail, and yield a performance equivalent to that of random assignment.

6 Conclusion

In this paper, we proposed a structural loss function which exploits the global neighborhood of a node in the network. The proposed loss function introduced a novel way of penalizing the boundary nodes. We have shown the effectiveness of the proposed structural loss function by combining it with the conventional ridge and laplacian regularizers. We also propose and exploit a homophily regularizer for a smooth transition in the discriminative behavior of nodes lying in dense parts of the network. The proposed regularization scheme helps in highlighting the discriminative behavior of nodes. We found that our stochastic gradient descent based learning algorithm converges in a few hundred iterations. Further, we observed that our approach outperforms all other approaches reasonably.

References

Baluja, S.; Seth, R.; Sivakumar, D.; Jing, Y.; Yagnik, J.; Kumar, S.; Ravichandran, D.; and Aly, M. 2008. Video suggestion and discovery for youtube: taking random walks through the view graph. In *Proceedings of the 17th international conference on World Wide Web*, 895–904. ACM.

Bishop, C. 2006. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer.

Callut, J.; Françoisse, K.; Saerens, M.; and Dupont, P. 2008. Semi-supervised classification from discriminative random walks. In *Machine learning and knowledge discovery in databases*. Springer. 162–177.

Chakrabarti, S.; Dom, B.; and Indyk, P. 1998. Enhanced hypertext categorization using hyperlinks. In *ACM SIGMOD Record*, volume 27, 307–318. ACM.

Grover, A., and Leskovec, J. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 855–864. ACM.

Jensen, D.; Neville, J.; and Gallagher, B. 2004. Why collective inference improves relational classification. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 593–598. ACM.

Kandola, J.; Cristianini, N.; and Shawe-taylor, J. S. 2002. Learning semantic similarity. In *Advances in neural information processing systems*, 657–664.

Kondor, R. I., and Lafferty, J. 2002. Diffusion kernels on graphs and other discrete structures. In *Proceedings of the 19th international conference on machine learning*, 315–322.

Lancichinetti, A.; Fortunato, S.; and Radicchi, F. 2008. Benchmark graphs for testing community detection algorithms. *Physical review E* 78(4):046110.

Leskovec, J.; Adamic, L. A.; and Huberman, B. A. 2007. The dynamics of viral marketing. *ACM Transactions on the Web (TWEB)* 1(1):5.

Lin, F., and Cohen, W. W. 2010. Semi-supervised classification of network data using very few labels. In *2010 International Conference on Advances in Social Networks Analysis and Mining*, 192–199. IEEE.

Lu, Q., and Getoor, L. 2003. Link-based classification. In *Proceedings of the 20th International Conference on Machine Learning (ICML)*, 496–503.

Macskassy, S. A., and Provost, F. 2003. A simple relational classifier. In *Proceedings of the Second Workshop on Multi-Relational Data Mining (MRDM-2003) at KDD-2003*.

Macskassy, S. A.; Provost, F.; and Provost, F. 2005. Netkit-srl: A toolkit for network learning and inference. In *Proceeding of the NAACSOS Conference*.

McAuley, J.; Targett, C.; Shi, Q.; and van den Hengel, A. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 43–52. ACM.

Nandanwar, S., and Murty, M. 2016. Structural neighborhood based classification of nodes in a network. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1085–1094. ACM.

Page, L.; Brin, S.; Motwani, R.; and Winograd, T. 1999. The pagerank citation ranking: Bringing order to the web. In *Stanford InfoLab*.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 701–710. ACM.

Sen, P.; Namata, G. M.; Bilgic, M.; Getoor, L.; Gallagher, B.; and Eliassi-Rad, T. 2008. Collective classification in network data. *AI Magazine* 29(3):93–106.

Smola, A. J., and Kondor, R. 2003. Kernels and regularization on graphs. In *Learning theory and kernel machines*. Springer. 144–158.

Sokolova, M., and Lapalme, G. 2009. A systematic analysis of performance measures for classification tasks. *Information Processing & Management* 45(4):427–437.

Talukdar, P. P., and Crammer, K. 2009. New regularized algorithms for transductive learning. In *Machine Learning and Knowledge Discovery in Databases*. Springer. 442–457.

Tang, L., and Liu, H. 2009. Relational learning via latent social dimensions. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 817–826. ACM.

Tang, L., and Liu, H. 2011. Leveraging social media networks for classification. *Data Mining and Knowledge Discovery* 23(3):447–478.

Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; and Mei, Q. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, 1067–1077. International World Wide Web Conferences Steering Committee.

Wang, X., and Sukthankar, G. 2013. Multi-label relational neighbor classification using social context features. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 464–472. ACM.

Zhou, D.; Bousquet, O.; Lal, T. N.; Weston, J.; and Schölkopf, B. 2004. Learning with local and global consistency. *Advances in neural information processing systems* 321–328.

Zhu, X., and Ghahramani, Z. 2002. Learning from labeled and unlabeled data with label propagation. Technical Report CMU-CALD-02-107, Carnegie Mellon University.