A Model of Random Walks on Bipartite Graph for Classification Problems

Abstract

We introduce a general framework of semi-supervised graphical model that is applicable to any classification tasks that can be represented with norminal features, requiring only a handhul of labeled examples. We conducted experiments on three different NLP tasks: prepositional phrase attachment, named entity classification, and domain-specific terminology extraction. We show both theoretically and empirically how the graph structure helps making use of large amount of unlabeled data. Based on the empirically study of underlying correlations between graph structures and classification performance, we incorporate active learning techniques to achieve the best learning rate with minimum requirement on labeled data.

1 Introduction

Semi-supervised learning is favored by its ability of effectively utilizing unlabeled data. Especially in NLP applications, labeled data usually requires expensive manual annotation work from linguistics experts, while on the contrary, large amount of unlabeled documents and text are easily acquired from the web. Among others, semi-supervised learning on graphs is a direction that has drawn great attentions from NLP researchers.

In this paper, we first have an overview of various semi-supervised graph methods in section **??**. In section **??**, we introduce a model that presents novelty in its bipartite graph structure and active learning capability. Section **??** describes the experiments on three datasets for different NLP tasks. We empirically evaluate the model and compare with other approaches. We look at other work that apply semi-supervised graph methods to NLP appications in section **??**.

2 An Overview of Semi-Supervised Learning on Graphs

Semi-supervised learning uses both labeled instances and unlabeled instances as training data. Typically the unlabeled data size is much larger than labeled data size. The learning aims at predicting the labels for unlabeled data.

2.1 Graph Construction

Graph-based SSL represents the labeled and unlabeled instances[1] jointly on a graph as nodes. Suppose there is a set of n examples, among which l are labeled, u are unlabeled; $n = l + u$, usually $n >> l$.[2] Denote L the labeled example set and U the unlabeled example set.

$L := \{(x_1, y_1), (x_2, y_2), ..., (x_l, y_l)\}$;
$U := \{x_{l+1}, x_{l+2}, ...x_{l+u}\}$.

Edges are usually undirected, representing the similarity of two instances. Denote the edge weight connecting two nodes x_i and x_j as w_{ij}, there can be several types of graph encoding the edge weights:

Fully connected graph. Eedge weight is defined by the Gaussian kernel, also known as Radial Basis Function (RBF) kernel:

$$w_{ij} = exp(-(\|x_i - x_j\|^2/2\sigma^2)$$

The edge weight decreases as the Euclidean distance of the two nodes increases, with rate depending on the parameter σ.

ϵNN graph. ϵ is the threshold distance value for two nodes to be connected by an edge. In the unweighted setting, $w_{ij} = 1$ if $\|x_i - x_j\| \leq \epsilon$ and 0 otherwise .

kNN graph. Every node links to the k nearest neighbors in Euclidean distance. The edges can either be unweighted or weighted using the Gaussian

[1]We use the term “instance” and “example” interchangabally in this paper.

[2]Notations introduced in this section will be used throughout the paper.

kernel function defined above. The common definition of a kNN graph connects i, j if the kNN relationship exists in at least one direction.

Apart from the choice of weighting, a more substantial pre-requisite for all graph-based learning methods is a well chosen feature space and an effective similarity function, so that the ”neighborhoods” deduced from the similarity or distance metric are meaningful. This is however, often difficult to practice in NLP tasks because of the discrete or heterogeneous feature spaces. How to select features and possibly project the features into a subspace is still an under-investigated problem (Alexandrescu and Kirchhoff, 2007).

2.2 Mincut

Blum and Chawla first formulated an SSL algorithm as a graph cut problem in (Blum and Chawla, 2001). In the binary classification case, each positive labeled instance acts as “source” and negative labeled instance acts as “sink”. A cut is defined as a set of edges whose removal blocks the flow from “sources” to “sinks”. The Mincut algorithm aims at finding the cut whose edge weight sum is the minimum. Utilizing the Ford-Fulkerson method for computing Max Flow, it is solvable in polynomial time. The learned predictor assigns positive label to every unlabeled node that ends up in the same component with positive labeled examples, and vice versa.

Mincut is based on a Marcov random field assumption. Edge weights represent the Markov transition probability between nodes, thus similar nodes have higher edge weights. The binary label case is equivalent to Boltzmann machine. Formalizing Mincut as a regularized risk minimization problem, we minimize the sum of a loss function and a regularizer. Suppose binary labels 1 and 0, the loss function is

$$\infty \sum_{i \in L} (y_i - f(x_i))^2$$

The infinity weight forces the labeled data to be fixed at their given labels. The regularizer is

$$\frac{1}{2} \sum_{i,j=1}^{u+l} w_{ij} (f(x_i) - f(x_j))^2, \text{where } f(x_i) \in \{0, 1\} \tag{1}$$

corresponding to the cut size.

Variants of Mincut

There are two drawbacks for the plain Mincut method. First, the infinity weight in loss function forces a “hard cut” – labeled data are fixed at the original labels, unable to handle noises in labeled data. Second, there’s no guarantee for balanced cut in sense of resulting class size. When multiple minimum cuts exist, the algorithm always returns the first solution, which is often not the optimal one from the practice perspective.

To address these problems, (Blum et al., 2004) extend the plain Mincut by randomizing the graph structure. They disturb the graph by adding artificial random noise to edge weights, then solve the Mincut on resulting graphs.

RatioCut (Hagen and Kahng, 1992) and NCut (Shi and Malik, 2000) circumvent the problem of unbalanced cut. Ratiocut measure class balance in terms of class size, while NCut balances edge weights.

$$RatioCut(A, B) = \frac{\sum_{i \in A, j \in B} w_{ij}}{|A|} + \frac{\sum_{i \in B, j \in A} w_{ij}}{|B|} \tag{2}$$

$$NCut(A, B) = \frac{\sum_{i \in A, j \in B} w_{ij}}{\sum_{i \in A} d_i} + \frac{\sum_{i \in B, j \in A} w_{ij}}{\sum_{i \in B} d_i}, \tag{3}$$

where the degree of node x_i is defined as $d_i = \sum_{j=i}^{n} w_{ij}$.

2.3 Harmonic Function and Gaussian Fields

(Zhu et al., 2003) applies Gaussian random fields on a graph consisting of labeled and unlabeled instances. Edges are weighted using the RBF kernel. The method shares the same assumption with other graph-based methods like nearest neighbor and mincuts, that similar examples should belong to same class. The difference is the Gaussian field is a continuous space, rather than discreet space over the unlabeled nodes.

If we express the regularization using the loss function and regularizer, we find it has the same loss function as Mincut,

$$\infty \sum_{i \in L} (y_i - f(x_i))^2$$

which fixes labeled data at given labels. The regularizer (equation 4) is based on the graph combinatorial Laplacian L 2.4), with an important relaxation that $f_i \in (0, 1)$, instead of $f_i \in \{0, 1\}$ as in Mincut.

$$\frac{1}{2}\sum_{i,j=1}^{u+l} w_{ij}(f(x_i) - f(x_j))^2 = f^T L f \qquad (4)$$

The solution to the problem is a harmonic function that has a closed form solution easily computed by matrix calculation. It also has the "harmonic" property as in equation 5.

$$f(j) = \frac{1}{\sum_i w_{ij}} \sum_{i \in Neighbor(j)} w_{ij} f(i), \qquad (5)$$

$$\text{for } j = l+1, l+2, ..., l+u$$

This provides the theoretical support that iterative label propagation algorithm converges to the harmonic function solution.

Connection to random walks

On the same graph, a random walker starts from node x_i. We can define the probability of stepping to node x_j in the next move be the normalized weight w_{ij}. If we assign 0 and 1 to the two classes of labeled examples, the harmonic function's solution at node x_i: $f(x_i)$ is the probability for a random walker starting from x_i and hitting an example labeled 1 before hitting a 0.

2.4 Spectral Methods

Spectral clustering is an unsupervised graph method. The central role of graph Laplacian connects several methods in previous section from the spectral theory view. We first look at the formal definitions of normalized, unnormalized graph Laplacian and their properties. The rest of the section discusses some points of view that connect spectral clustering with other graph methods.

Graph Laplacian

We will be using the following notations to describe an undirected graph $G = (V, E)$:
The node set $V = x_1, ..., x_n$;
The adjacency matrix $W = (w_{ij})_{i,j=1,...,n}$;
The degree of x_i, $d_i = \sum_{j=i}^{n} w_{ij}$;
The degree matrix D is the diagonal matrix with degrees $d_1, ..., d_n$ on the diagonal;

The unnormalized graph Laplacian is defined as

$$L = D - W$$

For every function $f : [n] \rightarrow R$,

$$f^T L f = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(f(x_i) - f(x_j))^2 \qquad (6)$$

It's trivially proved that L is positive semi-definite and symmetric. It follows that L has n non-negative eigenvalues, $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$. We denote their corresponding eigenvectors $\phi_1, ..., \phi_n$. The spectral decomposition is $L = \sum_{i=1}^{n} \lambda_i \phi_i \phi_i^T$.

An important property used in spectral clustering is that the number of zero eigenvalues equals the number of connected components in the graph.

The normalized graph Laplacian is defined as

$$\mathcal{L} = \mathbf{I} - D^{-12} W D^{-12}$$

For every function $f : [n] \rightarrow R$,

$$f^T \mathcal{L} f = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(\frac{f(x_i)}{\sqrt{d_i}} - \frac{f(x_j)}{\sqrt{d_j}})^2 \qquad (7)$$

The properties of L discussed above apply to $\mathcal{L}$.

Graph Laplacian in SSL is used to encode the smoothness of function f. For example, $f^T L f$ can be used as a regularizer in transductive classification since a low value of $f^T L f$ requires f changing very little in high density regions. In other words, $f^T L f$ makes a "cluster assumption" that decision boundary of a classifier lies in low density regions.

Spectral clustering algorithm

The spectral clustering works as follows: From the first k eigenvectors of L, $\phi_1, ..., \phi_k$, we span a new space $V \in R^{n \times k}$ which contains the k eigenvectors as columns. Then the data points are mapped to the rows of V, i.e., for $i = 1, ..., n$, the projection of data point x_i is the i-th row of V. We then perform clustering using traditional methods such as k-means on the new projections of data points.

Connection with Graph Cut

Recall that Mincut can be interpreted as a problem of minimizing the objective function (1) with the constraints of f being discrete labels and consistent with clamped labels.

Consider the RatioCut objective function (2), we define f

$$f(x_i) = \begin{cases} \sqrt{|B|/|A|} & \text{if } x_i \in A \\ \sqrt{|A|/|B|} & \text{if } x_i \in B \end{cases} \tag{8}$$

and thus can rewrite

$$f^T L f = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij} (f(x_i) - f(x_j))^2 \tag{9}$$

$$= |V| \cdot RatioCut(A, B) \tag{10}$$

If we relax the constraints f being a binary function to real numbers, we are able to use the above equation to approximate the problem of minimizing $RatioCut(A, B)$ to minimizing $f^T L f$ subject to constraint (8), which is equivalent to $f \perp 1$ and $\|f\| = \sqrt{n}$. The solution of the relaxed problem is the second eigenvector of L. The real-valued solutions can easily be transformed to labels or classes by taking threshold 0.

Following similar lines of calculations we can show that relaxed Normalized Ncut defined by (3) can be approximated by normalized spectral clustering (Shi and Malik, 2000).

Connection with Random Walk

From the random walk point of view, we define the transition matrix P as

$$P = D^{-1} W$$

The stationary probability $\pi_i = d_i / \sum_{j,k=1}^{n} w_{jk}$ is unique if the graph is connected and non-bipartite. For disjoint subset A,B, denote $P(B|A)$ as the the probability of a random walk starting with a node in A ends at a node in B. It has been proved that

$$Ncut(A, B) = P(A|B) + P(B|A)$$

. We thus can interprete the problem of minimizing Ncut equivalently as a partition of the graph such that random walks seldom transit across different classes. This consequently relates random walks to the normalized graph Laplacian.

Another concept that connects random walks with graph Laplacian is the commute distance. The commute distance is the expected time it takes for a random walker starting at x_i to travel to x_j and then back to x_i. Nodes are closer if there exist multiple paths so the commute distances are shorter in dense regions. The commute distance can be computed from the generalized inverse of the graph Laplacian L (Klein and Randic, 1993).

3 The Bipartite Graph Model

3.1 Graph Construction

We first formulate a classification task with the following parameters:

a. A set of n examples, among which l are labeled, u are unlabeled; $n = l + u$, usually $n >> l$. The l labeled examples and their labels form a labeled set $L := \{(x_1, y_1), (x_2, y_2), ..., (x_l, y_l)\}$;

b. The u unlabeled examples form an unlabeled set $U := \{x_{l+1}, x_{l+2}, ...x_{l+u}\}$;

c. Each example's feature is a subset of the norminal feature set $F = \{f_1, f_2, ..., f_m\}$. Write $x_i = (x_i^1, x_i^2, ..., x_i^m)$, and $x_i^j = 1$ if example x_i has nominal feature f_j, otherwise $x_i^j = 0$.

The goal is to predict the labels of any (all) unlabeled examples.

To construct the bipartite graph $G = (X, F, E)$ (X, F are node sets and E is edge set), we create examples nodes for each example $X = \{x_1, x_2, ..., x_n\}$, and feature nodes for each nominal feature $F = \{f_1, f_2, ..., f_m\}$. We connect an example node and a feature node using an undirected edge if and only if the feature is present in the example, $(x_i, f_i) \in E$ iff $x_i^j = 1$.

3.2 Random Walk

// Describe the random walk hitting time method.
// add algorithm

3.3 Label Propagation

An alternative impelmentation that solves the same problems as performing random walks is label propagation.

// add algorithm of tumbl

// explain or prove the equivalence of the two algorithms theoretically

// cite Zhu's 2002 paper

4 Active Learning

Observations in a pilot experiment on the preporsitional phrase attachment dataset motivated applying active learning. In this section, we first briefly describe the PP attachment classification task, dataset, and results form the pilot experiment. Then we present the active learning algorithm.

4.1 PP Attachment Dataset

1) ppattach disambiguation

2) dataset

3) state-of-art, backoff method

4.2 Pilot Experiment

use basic tumbl model

plot: learning curve from 10 to 1000 training size

plot: uncertianty - when sample size is small (e.g. 10 , 50) performance

variance is large, motivate a method to better choose examples to be labeled

4.3 Algorithm with Active Learning

// need to fill in after experiments

5 Experiments

5.1 PP attachemnt dataset

1) baseline: backoff method

2) tumbl, randomly pick labeled examples

3) active learning

Todo: describe baseline, describe experiment settings

plot accuracy of 3 methods with different training size

5.2 Named Entity Classification

1) describe task and dataset

2) describe baseline DLCoTrain

3) experiment with NEC data

4) experiment DLCoTrain and tumbl on ppattach set

5) analysis

5.3 AAN Terminology Extraction

Not sure about this, if time is limited, show only preliminary results, no comparison w/ other methods.

6 Related Work

In this section we review the related work where semi-supervised graphical models are successfully applied to various NLP tasks.

Sentence subjectivity classification using Mincut

(Pang and Lee, 2004) applied Mincut to a graph constructed of individual sentences as nodes to classify whether a sentence is subjective or objective. Only two nodes s and t are labeled initially, representing hte "subjective" class and "objective" class. They assign edge weight $w_{ij} = assoc(x_i, x_j)$ for all pairs of sentence nodes and weights $ind1(x_i)$, $ind2(x_i)$ to the edges connecting x_i and the s, t nodes.

The notation $assoc(x_i, x_j)$ (short for "association") is a similarity metric of how likely the two sentences will be in the same subjectivity class. In the experiments it is calculated by a *proximity* function of two sentences from a same document. $ind1(x_i)$, $ind2(x_i)$ (short for "individual") for sentence x_i are non negative possibilities that the sentence belongs to the two subjectivity classes. The scores can be priorly obtained from a first-pass classifier using sentence features and employing linguistic knowledge of sentiment indicators. The authors use unigram features to train a Naive Bayes classifier as the first-pass classifier. They then run a Mincut algorithm on the constructed graph and obtain the subjectivity classes for individual sentences.

Word Sense Disambiguation (WSD) using label propagation

It has been shown that iterative label propagation (LP) algorithm converges to the harmonic function. (Niu et al., 2005) implement an LP classifier for the WSD task. Their graph is constructed from labeled and unlabeled examples drawn from the standard SENSEVAL-3 task set and two other benchmark corpora. The weighting of the graph is calculated using contextual features, including part-of-speech of neighboring words with position information, unordered single words in topical context, and local collocations (Niu et al., 2005). The graph is built as a kNN graph, namely, two nodes are connected if either one is within the k nearest neighbor of the other, by either the cosine or the JS divergence distance metric.

On the same WSD task utilizing the LP algorithm, (Alexandrescu and Kirchhoff, 2007) proposed a data-driven approach for graph construction. It learns a projection from initial feature vector to a subspace. The new graph construction method is proved to be improving performance of the LP classification for WSD task.

Sentiment classification using label propagation and Mincut

(Rao and Ravichandran, 2009) build a lexical graph of unlabeled and labeled nodes, representing words that can be either positive or negative. word that can be either positive or negative. Edges represent some semantic relatedness that can be constructed using resources like WordNet or other thesaurus. They evaluate two semi-supervised methods: Mincut (including its variant randomized Mincut) and label propagation on the word sentiment classification task.

(Blair-goldensohn et al., 2008) use a similar label propagation method on a lexical graph built from WordNet, where a small set of words with known polarities are seeds. They predict sentiment of more words on the lexical graph to augment the sentiment word lexicon. (Brody and Elhadad, 2010) use label propagation over a graph constructed of adjectives to classify sentiments.

Multi-document summarization using random walks

Lexrank (Erkan and Radev, 2004) is a graph-based model for calculating sentence salience and has been applied to multi-document summarization. The graph consists of sentence nodes and edges represent the similarity of sentences, which can be cosine similarity with a cutoff threshold, or other distance measures. A salient sentence should share similar information with many sentences and is favored to be selected into the summary. Lexrank adopts the damping factor notion used in PageRank (**?**), which simulates a small probability of randomness (jumping to a random node instead of following the distribution of edges).

7 Conclusion

References

Andrei Alexandrescu and Katrin Kirchhoff. 2007. Data-driven graph construction for semi-supervised graph-based learning in nlp. In *HLT-NAACL*, pages 204–211.

Sasha Blair-goldensohn, Tyler Neylon, Kerry Hannan, George A. Reis, Ryan Mcdonald, and Jeff Reynar. 2008. Building a sentiment summarizer for local service reviews. In *In NLP in the Information Explosion Era*.

Avrim Blum and Shuchi Chawla. 2001. Learning from labeled and unlabeled data using graph mincuts. In *ICML*, pages 19–26.

Avrim Blum, John D. Lafferty, Mugizi Robert Rwebangira, and Rajashekar Reddy. 2004. Semi-supervised learning using randomized mincuts. In *ICML*.

Samuel Brody and Noemie Elhadad. 2010. An unsupervised aspect-sentiment model for online reviews. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 804–812, Los Angeles, California, June. Association for Computational Linguistics.

Günes Erkan and Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Intell. Res. (JAIR)*, 22:457–479.

Lars W. Hagen and Andrew B. Kahng. 1992. New spectral methods for ratio cut partitioning and clustering. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 11(9):1074–1085.

D. Klein and M. Randic. 1993. Resistance distance.

Zheng-Yu Niu, Dong-Hong Ji, and Chew Lim Tan. 2005. Word sense disambiguation using label propagation based semi-supervised learning. In *ACL*.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *ACL*, pages 271–278.

Delip Rao and Deepak Ravichandran. 2009. Semi-supervised polarity lexicon induction. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 675–682, Athens, Greece, March. Association for Computational Linguistics.

Jianbo Shi and Jitendra Malik. 2000. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905.

Xiaojin Zhu, Zoubin Ghahramani, and John D. Lafferty. 2003. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, pages 912–919.