Uncertainty Aware Graph Gaussian Process for Semi-Supervised Learning

Zhao-Yang Liu,[1] Shao-Yuan Li,[1] Songcan Chen,[1] Yao Hu,[2] Sheng-Jun Huang*[1]
[1]College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics
Collaborative Innovation Center of Novel Software Technology and Industrialization Nanjing 211106, China
[2]YouKu Cognitive and Intelligent Lab, Alibaba Group, Hangzhou, China
{zhaoyangliu, lisy, s.chen, huangsj}@nuaa.edu.cn, yaoohu@alibaba-inc.com

Abstract

Graph-based semi-supervised learning (GSSL) studies the problem where in addition to a set of data points with few available labels, there also exists a graph structure that describes the underlying relationship between data items. In practice, structure uncertainty often occurs in graphs when edges exist between data with different labels, which may further results in prediction uncertainty of labels. Considering that Gaussian process generalizes well with few labels and can naturally model uncertainty, in this paper, we propose an Uncertainty aware Graph Gaussian Process based approach (UaGGP) for GSSL. UaGGP exploits the prediction uncertainty and label smooth regularization to guide each other during learning. To further subdue the effect of irrelevant neighbors, UaGGP also aggregates the clean representation in the original space and the learned representation. Experiments on benchmarks demonstrate the effectiveness of the proposed approach.

Introduction

Datasets with graph or network structures are all around in wide applications. It is often desirable to predict the labels for nodes in the graph. For example, in a social network, we may be interested in predicting the profession of individuals; in a citation network, we might want to predict the topics of documents. Sometimes, only a small number of labeled examples are available, and it is costly to collect a large set of labeled data for training. In such cases, making good use of the graph structure which contains information about the relationship between data items is critical. Such learning problem is referred as graph-based semi-supervised learning (GSSL)(Zhu 2005).

During the past years, various studies have been conducted for GSSL. Early work led to the development of explicit graph-based regularization(Zhu, Ghahramani, and Lafferty 2003; Belkin, Niyogi, and Sindhwani 2006; Zhang et al. 2010), which rely on the label smooth assumption that data connected to each other in the graph tend to share the same label. Recently, implicit graph regularization via learned node representation have emerged, including graph embedding works(Perozzi, AI-Rfou, and Skiena 2014; Yang, Cohen, and Salakhutdinow 2016) and the powerful graph convolutional networks (GCN)(Wu et al. 2019; Kipf and Welling 2017).

In real world tasks, the graph structure is not always precisely given, and usually suffer from uncertainty issues. For example, it is a common case that edges connecting nodes with different labels appear in a graph. In other words, the connected neighbors could be irrelevant to the node. This in turn would lead to uncertainty on the predictions of the learning model. It is thus a significant challenge to enhance the learning model with ability to handle structure uncertainty in the graph. Attempts for GCN have been made towards this target. The approach in (Vashishth et al. 2019) added the label prediction uncertainty of nodes as parameters and optimized them jointly with the network parameter by back propagation. The method in (Zhang et al. 2019) treated the graph structure as one realization of a parametric family of random graphs and proposed a Bayesian GCN to solve it.

While the graph neural networks have generally surpassed Laplacian and embedding based methods in predictive performance, they require a relatively large set of labeled validation set to prevent over-fitting, whereas the labeled data is usually limited in many applications. To bridge the gap between the simpler models and the more data intensive graph neural networks, Gaussian process model for GSSL is proposed in (Ng, Colombo, and Silva 2018), which is data-efficient and can achieve competitive performance with GCN when there are sufficient labeled data. However, this method does not consider the uncertainty issue during the model inference and prediction.

In this paper, considering the appealing properties of Gaussian process, which generalize well with few labels and can naturally incorporate the uncertainty of inferences in the learning process, we propose an Uncertainty aware Graph Gaussian Process based approach (UaGGP) for GSSL. We utilize the prediction uncertainty and label smooth regularization to guide each other during the learning process. To further alleviate the possible side effect of irrelevant neigh-

*This research was supported by NSFC(61732006, 61572252), the Aerospace Power Funds No. 6141B09050342, the Fundamental Research Funds for the Central Universities, NO. NE2019104 and Collaborative Innovation Center of Novel Software Technology and Industrialization. Sheng-Jun Huang is the corresponding author.

borhoods on the learned node representation, we also propose to aggregate the clean data representation in the original input space and the learned representation. We examine the resilience of the proposed approach on three graph structured benchmark datasets.

Related Work

Researchers have developed various of technologies for graph-based semi-supervised learning during the past decades. Laplacian regularization and graph embedding are two important ways. And recently, methods based on neural networks become more popular.

In graph-based semi-supervised learning, the graph Laplacian regualrization is employed (Zhu, Ghahramani, and Lafferty 2003; Belkin, Niyogi, and Sindhwani 2006; Weston, Ratle, and Collobert 2008) by combining with different types of loss functions. For example, in (Zhu, Ghahramani, and Lafferty 2003), the authors defined a Gaussian Random Field (GRF) on the graph, and optimized the soft labels of unlabeled points by minimizing the quadratic energy function. The manifold regularization (Belkin, Niyogi, and Sindhwani 2006) has a similar idea for learning from labeled and unlabeled examples by taking the adjacent relations as the data geometry. Besides, some research works (Gadde and Ortega 2015; Chen and Wei 2018) in graph signal processing often consider the target on graph as a signal vector to recover (Gadde, Anis, and Ortega 2014; Chen et al. 2015). If it is assumed that the signal vector elements obey a multivariate Gaussian prior distribution, the reconstruction of the signal is equal to the MAP inference.

In order to seek more effective feature representation with lower dimensions, various methods are proposed based on graph embedding(Cai, Zheng, and Chang 2018). DeepWalk (Perozzi, AI-Rfou, and Skiena 2014) learns a representation which encodes structural regularities by using local information from truncated random walks. The advantage of DeepWalk is that, the learned representations are independent of models, which thus can be used in any classification method. The method in (Yang, Cohen, and Salakhutdinow 2016) combines embedding learning with graph-based semi-supervised learning, which utilizes the learned embedding and input feature together to predict both the class label and graph context.

In recent years, GNN (Zhang, Cui, and Zhu 2018; Zhou et al. 2019) has been attracting more and more attentions. GCN methods (Kipf and Welling 2017; Defferrard, Bresson, and Vandergheynst 2016) deal with GSSL in the view of deep learning via the spectral theory and achieve amazing results compared with traditional spectral approaches. However, GCN can not provide the confidence score for the predictions. It is desirable to know how much to trust the model output on many occasions. (Vashishth et al. 2019) tries to provide prediction distribution for each unlabeled node by setting the pseudo label mean score and variance as model parameters, which are optimized by back propagation. Whereas in our work based on GP, confidence scores are inferred naturally, and thus the label smooth constraint can be incorporated into the learning objective of the Graph Gaussian Processes.

Background

In this section, we briefly introduce the necessary background basis for Gaussian Process learning and the notations used in the paper.

Gaussian Processes

In 2006, Rasmussen and Williams (2006) presented a comprehensive tutorial on Guassian Processes (GPs) and their natural applications in machine learning. Formally a GP is defined as a collection of random variables, of which any finite subsets have joint Gaussian distributions. Thus a GP can be fully specified by its mean function $m(x)$ and covariance function $k(x, x')$:

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')). \tag{1}$$

Defined over functions, GPs are natural generalizations of Gaussian distribution over vectors. Used as priors over real-valued functions $f(x)$, GPs provide rich nonparametric models for the machine learning field. The prior is updated in light of the training examples to get the posterior, which is then used to make predictions for unseen test data.

Given a dataset of input $\mathbf{X} = \{x_i\}_{i=1}^n$ and their corresponding labels $\mathbf{y} = \{y_i\}_{i=1}^n$, GPs define a prior over the latent function value at the input examples:

$$p(f(x_i)) = \mathcal{N}(f(x_i)|m(x_i), k(x_i, x_i)). \tag{2}$$

The observed label y_i is connected to the function value via the conditional likelihood function:

$$y_i|f(x_i) \sim p(y_i|f(x_i)). \tag{3}$$

Therefore, the joint distribution of data and latent function becomes:

$$\begin{aligned} p(\mathbf{y}, \mathbf{f}) &= \prod_{i=1}^{n} p(y_i|f(x_i))p(f(x_i)) \\ &= \prod_{i=1}^{n} p(y_i|f(x_i))\mathcal{N}(\mathbf{f}|\mathbf{m}_n, \mathbf{K}_{nn}) \end{aligned} \tag{4}$$

Here $\mathbf{m}_n$ denotes the mean vector on $\mathbf{X}$, $\mathbf{K}_{nn}$ denotes the covariance matrix over all data pairs. In this paper, we concern classification tasks. For binary classification, the conditional likelihood is implemented as

$$p(y_i|f(x_i)) = \mathcal{B}(y_i|\pi(f_i)) = \pi(f_i)^{y_i}(1 - \pi(f_i))^{1-y_i}. \tag{5}$$

Here $\pi(x) = \int_{-\infty}^{x} \mathcal{N}(a|0, 1)\mathrm{d}a$ is the cumulative probability density function of the normal distribution. For the multi-class problems, $p(y_i|f(x_i))$ is implemented by the robust-max likelihood(Wu, Lin, and Weng 2004). The learning interest in GPs is to infer the posterior $p(\mathbf{y}|\mathbf{f})$ and the model parameters through maximizing the marginal likelihood $p(\mathbf{y})$.

Scalable GPs Learning

While GPs can be conveniently used to specify flexible learning functions, the inference for parameters and posterior is not easy. First, for cases where the likelihood $p(\mathbf{y})$ is non-Gaussian, e.g., in classification problems, the marginal

likelihood $p(\mathbf{y})$ and the posterior $p(\mathbf{f}|\mathbf{y})$ must be approximated. Second, the computation for inference often has a complexity of $\mathcal{O}(N^3)$ (see (Rasmussen and Williams 2006) for details), which limited their application to large data.

Recently, scalable inference algorithms are proposed(Snelson and Ghahramani 2005; Titsias 2009; Hensman, Matthews, and Ghahramani 2015). As we concern classification tasks in this paper, we review the KLSp approach proposed by(Hensman, Matthews, and Ghahramani 2015), which empirically outperformed state-of-the art methods and can be optimized using SGD, making the GP classification applicable to big data.

Objective function derivation The KLSp method is derived using variational approximation. As in (Snelson and Ghahramani 2005), a set of t inducing points $\mathbf{Z} = [z_t, ..., z_t]^\top$ are introduced. Their latent function values are represented as $\mathbf{u} = [f(z_1), ..., f(z_t)]$. Assuming the mean function value $m(x)$ is zero, the conditional GP of $\mathbf{f}$ over $\mathbf{u}$ can be derived as

$$p(\mathbf{f}|\mathbf{u}) = \mathcal{N}(\mathbf{K}_{nt}\mathbf{K}_{tt}^{-1}\mathbf{u}, \mathbf{K}_{nn} - \mathbf{Q}_{nn}), \tag{6}$$

where $\mathbf{K}_{tt}$ is the covariance matrix at pairs of the inducing points $\mathbf{Z}$, $\mathbf{K}_{nt}$ is the covariance matrix across the $\mathbf{X}$ and $\mathbf{Z}$, $\mathbf{Q}_{nn} = \mathbf{K}_{nt}\mathbf{K}_{tt}^{-1}\mathbf{K}_{nt}^T$.

Combing Eqs.3 and 6, the joint distribution of $\mathbf{y}$,$\mathbf{f}$ and $\mathbf{u}$ takes the form $p(\mathbf{y}, \mathbf{f}, \mathbf{u}) = p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{u})p(\mathbf{u})$. Following the standard derivation of variational inference, the Evidence Lower Bound(ELBO) for $p(\mathbf{y})$ is obtained:

$$\mathcal{L}_{LB} = \sum_{i=1}^{n} \mathbb{E}_{q(f(x_i))}[\log p(y_i|f(x_i))] - \mathrm{KL}[q(\mathbf{u})||p(\mathbf{u})]. \tag{7}$$

Here $q(\mathbf{f})$is defined as $q(\mathbf{f}) := \int p(\mathbf{f}|\mathbf{u})q(\mathbf{u})\mathrm{d}\mathbf{u}$.

Training With the prior of $\mathbf{u}$ defined as $p(\mathbf{u}) = \mathcal{N}(\mathbf{0}, \mathbf{K_{tt}})$ and the posterior to be a multivariate Gaussian distribution $q(\mathbf{u}) = \mathcal{N}(\mathbf{v}, \mathbf{S})$, the inducing points $\mathbf{Z}$, parameters $\mathbf{v}, \mathbf{S}$ of the posterior distribution $q(\mathbf{u})$, and the covariance function parameters are optimized by maximizing the ELBO in Eq. 7 using gradient based optimization and SGD for large data.

Prediction Using the inferred variational Gaussian distribution $q(\mathbf{u})$, the latent function value are calculated as follows:

$$\begin{aligned} p(\mathbf{f}_*|\mathbf{y}) &= \int p(\mathbf{f}_*|\mathbf{f}, \mathbf{u})p(\mathbf{f}, \mathbf{u}|\mathbf{y})\mathrm{d}\mathbf{f}\mathrm{d}\mathbf{u} \\ &\approx \int p(\mathbf{f}_*|\mathbf{f}, \mathbf{u})p(\mathbf{f}|\mathbf{u})q(\mathbf{u})\mathrm{d}\mathbf{f}\mathrm{d}\mathbf{u} \\ &= \int p(\mathbf{f}_*|\mathbf{u})q(\mathbf{u})\mathrm{d}\mathbf{u}. \end{aligned} \tag{8}$$

The optimization of our UaGGP approach is implemented by adapting the process of KLSp.

The UaGGP Approach

In the graph-based semi-supervised learning problem, in addition to a set of data points $\mathbf{X} = \{x_i\}_{i=1}^n$ and the labels for a subset of the data $\mathbf{y}_L = \{y_i\}_{i=1}^{n_L}$ with $y_i \in \{1, ..., K\}$, a graph structure $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is also available. A symmetric binary adjacency matrix $A \in \{0, 1\}^{n\times n}$ can be used to represent the relationship. The target is to predict the labels $\mathbf{y}_U = \{y_i\}_{i=1+n_L}^n$ for the remaining unlabeled data.

To make use of the graph structure in GPs, a Graph Gaussian Process (GGP) model is proposed in (Ng, Colombo, and Silva 2018), which introduced another latent function h, whose GP prior is defined as the average of the values of f over the 1-hop neighborhood Ne given by the adjacent matrix A:

$$g_i = \frac{f(x_i) + \sum_{k\in Ne(i)} f(x_k)}{1 + D_{i,i}}. \tag{9}$$

And the joint distribution is defined as

$$p(\mathbf{Y}, \mathbf{g}|\mathbf{X}, A) = p(\mathbf{g}|\mathbf{X}, A)\prod_{i=1}^{n} p(y_i|g_i). \tag{10}$$

In this paper, we extend the work of (Ng, Colombo, and Silva 2018) by considering the graph structure uncertainty issue. In practice, inexact graph structure are common, i.e., edges may occur between data with different labels, which in turn would result in misleading neighbors and predictions. We propose the Uncertainty aware Graph Gaussian Process based approach (UaGGP), which improves GGP from two perspectives. First, considering that GP can naturally incorporate uncertainty, UaGGP utilizes the prediction uncertainty and label smooth regularization to guide each other during learning. Second, considering that the GGP can be interpreted as learning Laplacian matrix transformed feature representation, to alleviate the possible side effect of irrelevant neighborhoods on learned representation, UaGGP further aggregates the clean data representation in the original input space and the learned representation. We explain the two points in the following subsections.

Regularization of Inference

From the definition of g Eq. 9, it can be seen that prior of g can be written as

$$p(\mathbf{g}|\mathbf{X}, A) = \mathcal{N}(\mathbf{0}, P\mathbf{K_{nn}}P^\top), \tag{11}$$

where $P = (I + D)^{-1}(I + A)$ is the random walk matrix. This means that implementing the GGP with the KLSp algorithm(Hensman, Matthews, and Ghahramani 2015) amounts to implementing a new covariance function, and the optimization process keeps the same.

Using the prediction function defined in Eq. 8, combined with the Gaussian form of $p(\mathbf{g}_*|\mathbf{u})$ and $q(\mathbf{u})$, the posterior mean and posterior covariance of $p(\mathbf{g}_*|\mathbf{y})$ can be got easily:

$$\mu = \hat{\mathbf{K}}_{nt}\hat{\mathbf{K}}_{tt}^{-1}\mathbf{v}, \tag{12}$$

$$\mathbf{\Sigma} = \hat{\mathbf{K}}_{nn} + \hat{\mathbf{K}}_{nt}\hat{\mathbf{K}}_{tt}^{-1}(\mathbf{S} - \hat{\mathbf{K}}_{tt})(\hat{\mathbf{K}}_{nt}\hat{\mathbf{K}}_{tt}^{-1})^\top, \tag{13}$$

where $\hat{\mathbf{K}} = P\mathbf{K}_{nn}P^\top$.

The mean and covariance predictions $\mu, \mathbf{\Sigma}$ for the data are utilized in this paper to conduct regularization. The idea is as follows. For an edge which lies between a pair of nodes v_i and v_j, the inferences of both nodes are supposed to be close, so we may define and minimize a distance $d(v_i, v_j)$ between both inferred results. However, when the model is

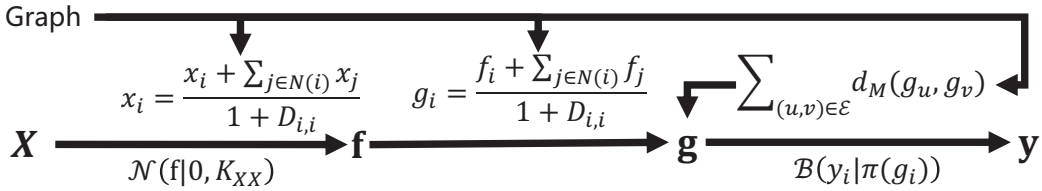


Figure 1: The sketch of the UaGGP: feature aggregation are conducted in both the input sapce and the Hilbert space. The label smooth regularization is applied to model predictions.

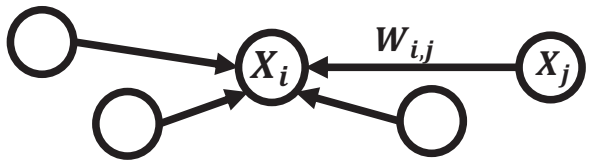


Figure 2: Feature Aggretation: One node can be represented by itself and its neighbour nodes, different methods have different aggreation weights.

not confident in the inference of v_i or v_j or both of them, the distance $d(v_i, v_j)$ should not be tight.

Following (Orbach and Crammer 2012; Vashishth et al. 2019), we take the symmetric Mahalanobis distance as the measure. Let μ_i, μ_j denote the latent function mean vector of nodes v_i and v_j respectively, diagonal matrices Σ_i and Σ_j denote the uncertainty of v_i and v_j respectively, where $\mu_i \in \mathbb{R}^K$ and $\Sigma \in \mathbb{R}^{K \times K}$. That is, $(\Sigma_i)_{kk}$ denotes the variance of $\mu_{i,k}$, which represents the score that node v_i belongs to label k. The distance between them would be expressed as

$$d(v_i, v_j) = (\mu_i - \mu_j)^\top (\Sigma_i^{-1} + \Sigma_v^{-1})(\mu_i - \mu_j). \quad (14)$$

One important property of the distance $d(v_i, v_j)$ is that if at least one of the two diagonal matrices has large eigenvalue, the constraint that expect μ_i and μ_j to be close will be relaxed. For any pair of nodes, if an edge exists between them, we enforce the GP inferences of them to be close, so the regularization term would be written as

$$\mathcal{L}_{LS} = \sum_{(i,j) \in \mathcal{E}} (\mu_i - \mu_j)^\top (\Sigma_i^{-1} + \Sigma_v^{-1})(\mu_i - \mu_j) \quad (15)$$

The objective learning function is defined as the sum of the ELBO in Eq. 7 and the label smooth regularization in Eq. 15, as follows:

$$\mathcal{L} = -\mathcal{L}_{LB} + \lambda \mathcal{L}_{LS} \quad (16)$$

Feature Aggregation

In Eq. 11, the kernel matrix $\mathbf{K}_{nn}$ can be written as $\mathbf{K}_{nn} = \Phi_n \Phi_n^\top$, where Φ denotes the feature map corresponding to the covariance function. Thus the GGP can be interpreted as aggregating nodes feature in the learned Hilbert space.

Considering that the learned node representation in the Hilbert space might be badly affected by the uncertainty in the graph structure and prediction degeneration, we proposed to also conduct aggregation of node features in the original reliable input space.

GCN(Kipf and Welling 2017) has suggested several ways to aggregate nodes feature considering graph neighborhood:

$$\hat{\mathbf{X}} = \begin{cases} A\mathbf{X} \\ (A+I)\mathbf{X} \\ (I+D)^{-1}(I+A)\mathbf{X} \\ (I + D^{-1/2}AD^{-1/2})\mathbf{X} \\ D^{-1/2}AD^{-1/2}\mathbf{X} \\ \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}\mathbf{X} \end{cases}$$

Here D denotes the diagonal degree matrix of the graph, $\tilde{A} = A + I$, $\tilde{D}$ is the degree matrix of $\tilde{A}$. As Figure 2 shows, various GCN methods aggregate feature by using different weights W_{ij} definition. In GCN(Kipf and Welling 2017), the renormalization trick gained best performance. In our work, for fair comparison with (Ng, Colombo, and Silva 2018), we take a weighted average of features and exploit the simple implementation $\hat{\mathbf{X}} = (I + D)^{-1}(I + A)\mathbf{X}$. This is similar as g_i, the new normalized representation $\hat{x_i}$ for each node is constructed utilizing the one-hop neighborhoods:

$$\hat{x_i} = \frac{x_i + \Sigma_{k \in Ne(i)} x_k}{1 + D_i}. \quad (17)$$

In UaGGP, the new data representations $\{\hat{x_i}\}$ are used as inputs for the learning model.

The sketch of UaGGP is shown in Figure 1, where the feature aggregation are conducted both in the original input space and the learned Hilbert space, and the soft label smooth regularization term is applied.

Optimization

The optimization of UaGGP can be easily implemented using the scalable variational inference algorithm of KLSp(Hensman, Matthews, and Ghahramani 2015). The variational parameters $\mathbf{Z}, \mathbf{v}, \mathbf{S}$ and covariance function parameters are jointly optimized by gradient based method.

Time Complexity

The computational complexity of UaGGP composes of three parts: 1) computing the regularization term $\mathcal{L}_{LS}$ in Eq. 15, which incurs $\mathcal{O}(|\mathcal{E}|)$; 2) computing the new representations $\{\hat{x_i}\}$ in Eq. 17, which incurs $\mathcal{O}(nD_{max})$ with D_{max} representing the maximum node degree; 3) the gradient based variational inference algorithm, which is $\mathcal{O}(n_L t^2)$ (Ng,

Table 1: Statistics of the datasets used in experiments for the classification task.

	Type	Nodes	Edges	Features	Classes	Label Rate	Label Mismatch	Test Nodes
Cora	Citation	2708	5429	1433	7	0.052	0.002	1000
Citeseer	Citation	3327	4732	3703	6	0.036	0.003	1000
Pubmed	Citation	19717	44338	500	3	0.003	0.0	1000

Table 2: Performance of methods on benchmark datasets. Results for baselines are taken from (Liao et al. 2018; Vashishth et al. 2019; Kipf and Welling 2017; Ng, Colombo, and Silva 2018).

Method	Cora	Citeseer	Pubmed
LP(Zhu, Ghahramani, and Lafferty 2003)	68.0%	45.3%	63.0%
ManiReg(Belkin, Niyogi, and Sindhwani 2006)	59.5%	60.1%	70.7%
SemiEmb(Weston, Ratle, and Collobert 2008)	59.6%	59.0%	71.1%
DeepWalk(Perozzi, AI-Rfou, and Skiena 2014)	67.2%	43.2%	65.3%
Planetoid(Yang, Cohen, and Salakhutdinow 2016)	75.7%	64.7%	77.2%
GCN(Kipf and Welling 2017)	81.5%	70.3%	**79.0%**
GGP(Ng, Colombo, and Silva 2018)	80.9%	69.7%	77.1%
UaGGP	**82.7%**	**70.7%**	76.7%

Colombo, and Silva 2018). In practice, the first and second computation terms are small because the graphs are typically sparse.

When implementing the variational inference algorithm, we choose the number of inducing points t as the number of labeled data n_L, which is small compared to the total number of examples. In fact, for large data, the algorithm can be easily conducted in parallel or in a stochastic manner by randomly selecting mini-batches of the data.

Experiments

Datasets and Baselines

We follow the exact experimental setup in(Ng, Colombo, and Silva 2018; Kipf and Welling 2017; Vashishth et al. 2019), where GCN and GPP are known to perform well. Described in Table 1, the three benchmark datasets are citation networks. Each node denotes a document represented by sparse bag-of-words features, each edge denotes the citation connection between two documents. The label of each document represents one topic. In Table 1, Label Rate denotes the percentage of labeled data, Label Mismatch denotes the proportion of edges that lie between different labels among the labeled data.

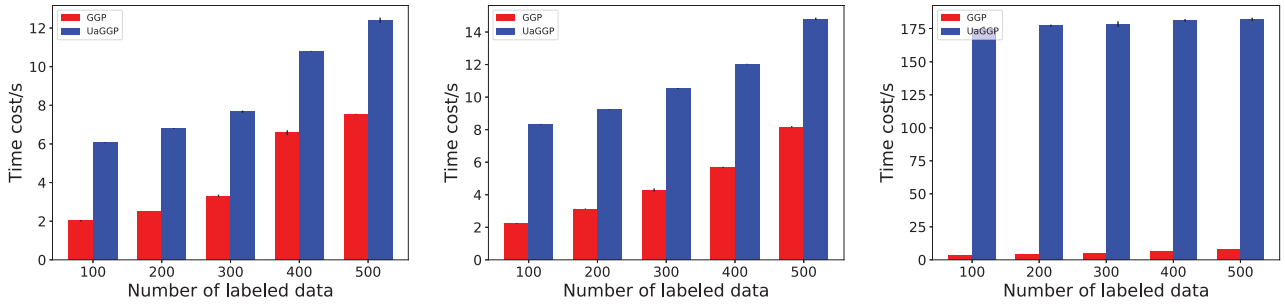
For fair comparison with (Ng, Colombo, and Silva 2018), we also use the 3rd degree polynomial kernel function for the covariance, and the joint gradient based optimization of variational parameters and model parameters are optimized using ADAM(Kingma and Ba 2015). The value of parameter λ is set to 0.001 for *Cora* and *Citeseer*, for *Pubmed*, λ=0.0001. All experiments are repeated for 10 times and the mean values are recorded.

We compare against several baselines including traditional methods, the well performed GCN(Kipf and Welling 2017) and GGP(Ng, Colombo, and Silva 2018).

- LP(Zhu, Ghahramani, and Lafferty 2003) defined a Gaussian random field model which propagates a node label to its neighbors by graph weights.
- ManiReg(Belkin, Niyogi, and Sindhwani 2006) built on a framework based a data-dependent regularization which exploits the geometry of the marginal distribution.
- SemiEmb(Weston, Ratle, and Collobert 2008) combined deep architectures with semi-supervised learning embedding to improve the unsupervised method.
- DeepWalk (Perozzi, AI-Rfou, and Skiena 2014) learned the latent representations of nodes by using local information obtained from truncated random walks.
- Palnetoid(Yang, Cohen, and Salakhutdinow 2016) learned an embedding for each node by jointly predicting the class label and the neighborhood context in the graph.
- GCN(Kipf and Welling 2017) presented one approach based on convolutional neural networks with outstanding performance.
- GGP(Ng, Colombo, and Silva 2018) is a Gaussian process based model which incorporates the graph neighborhoods into the GP prior.

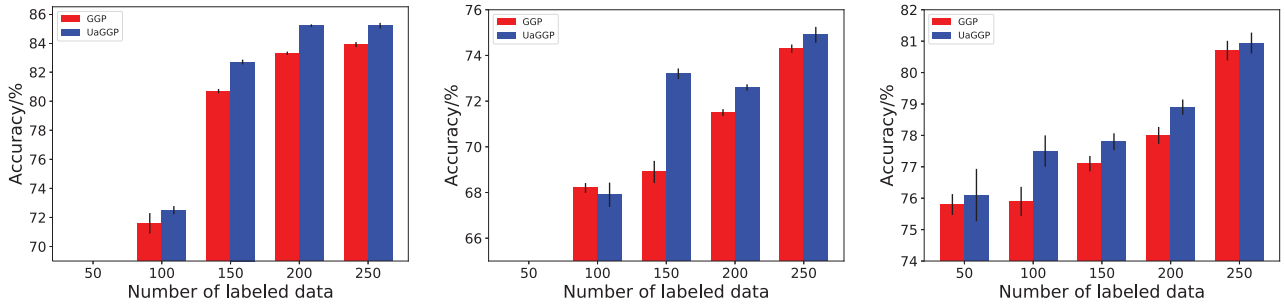
Comparison Results

The classification accuracy results of all methods for the three datasets are shown in Table 2. Due to the same experimental setup, results for the baselines are taken from (Liao et al. 2018; Vashishth et al. 2019; Kipf and Welling 2017; Ng, Colombo, and Silva 2018). As shown in Table 2, UaGPP consistently outperforms the baselines on *Cora* and *Citeseer*. Note that the original GGP is marginally worse than GCN, which uses additional 500 labeled data for early stopping. In the original GGP paper(Ng, Colombo, and Silva 2018), the authors also conducted one comparison 'GGP-X' which used the additional labels for training and obtained much better performance. Using only the training labels,



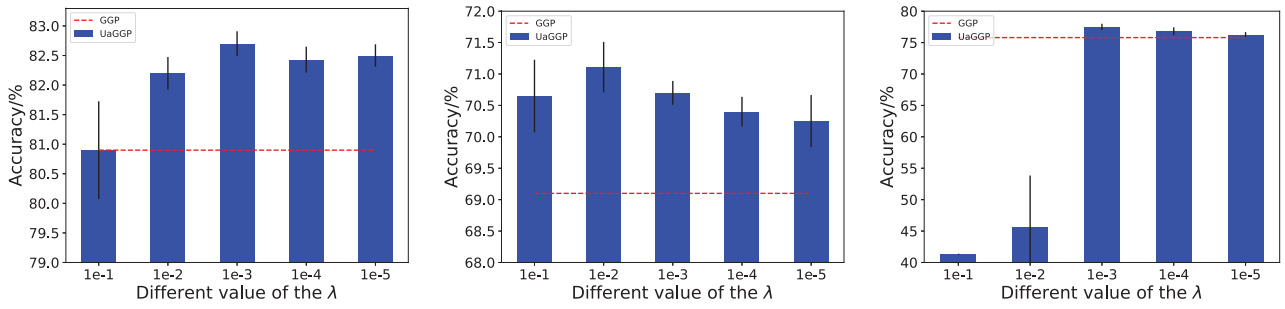
(a) Cora (b) Citeseer (c) Pubmed

Figure 3: Time cost per 100 iterations varies with different number of labeled nodes for three datasets.



(a) Cora (b) Citeseer (c) Pubmed

Figure 4: Performance variation over different number of labeled nodes. For GGP and UaGPP, the results not shown for $n_L = 50$ on *Cora* are 47.2 ± 0.44 vs 51.0 ± 0.34, results for $n_L = 50$ on *Citeseer* are 45.8 ± 2.26 vs 32.8 ± 1.61.



(a) Cora (b) Citeseer (c) Pubmed

Figure 5: Performance variation over different λ from $1e^{-1}$ to $1e^{-5}$. The red dotted line indicates the results of GGP(Ng, Colombo, and Silva 2018).

Table 3: Feature aggregation ablation study on GGP(Ng, Colombo, and Silva 2018) and UaGGP with three different aggregation manners: aggregation in the Hilbert space, the Euclidean space and in both spaces.

Space	Dataset	GGP	UaGGP
Hilbert	Cora	80.9%	82.3%
Euclidean		81.6%	82.4 %
Hi+Eu		**82.8**%	**82.7**%
Hilbert	Citeseer	69.7%	70.1 %
Euclidean		69.1%	70.2%
Hi+Eu		**70.7**%	**70.7**%
Hilbert	Pubmed	75.8%	76.5 %
Euclidean		**77.2**%	77.4 %
Hi+Eu		77.0%	**77.5**%

UaGPP beats GCN, which implies that our UaGPP makes better use of the data.

On *Pubmed*, UaGPP is marginally degenerated but competitive with GGP. We guess that this may be due to that *Pubmed* has very few labeled data, in which case the label smooth regularization term $\mathcal{L}_{LS}$ in Eq. 15 would not be helpful. Too few labels may even have side effects on $\mathcal{L}_{LS}$. In the next subsection, we explore the effect of label size and regularization strength on the learning performance.

We also examine the efficiency of the compared methods. We vary the number of nodes in $[100, 200, 300, 400, 500]$, and the results are shown in Figure 3.

Figure 3 shows how the time cost changes with the increase of labeled data size. It can be observed that the time cost of UaGGP is higher than GGP on the *Cora* and *Citeseer*. Especially UaGGP consumes much more time on the *Pubmed* dataset. Because the *Pubmed* contains much more nodes and edges, and all data points are need to be inferred by GP in each iteration and the symmetric Mahalanobis distance is calculated for each edge in the graph. So a large portion of time is consumed for the label smooth term. Note that the proposed approach can be easily implemented in parallel. The time cost may be significantly reduced with parallel implementation.

Performance Study with Varying n_L and λ

In this subsection, we explore the performance of UaGGP in settings with different number of labeled data and different strength of the regularization term $\mathcal{L}_{LS}$. We test the number of labeled data n_L in range of $[50, 100, 150, 200, 250]$, and set the value of λ as 0.0001 for $n_L = 50$ and 0.001 for other cases. We also report the performance of GGP for comparison.

The results with varying label size are shown in Figure 4. It can be observed that the proposed UaGGP approach consistently outperforms GGP with different number of labeled examples in most cases. A special case is on *Citeseer* when only 100 labeled examples are available, GGP achieves a slightly better performance than UaGGP. These results validate that the regularization term plays a positive role as long as labels are not too few.

Next, we vary the value of the parameter λ in the set $[10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}]$ to study the effect of the strength of regularization $\mathcal{L}_{LS}$. The number of labeled data n_L for *Cora* and *Citeseer* is the same as in Table 1. For *Pubmed*, $n_L = 100$ labels are used to ensure enough labels. The results are shown in Figure 5. It can be seen that the regularization term helps in most cases. On one hand, too large λ, e.g.,$\lambda > 1e-1$ on *Cora* and $\lambda > 1e-3$ on *Pubmed*, could enforces too strong regularization and results in performance collapse. On the other hand, So the strength of the regularization should be properly determined. too small λ could reduces the UaGGP to GGP and the results would not collapse but may get a little worse.

Feature Aggregation Study

As mentioned before, we can view the GGP method as conducting feature aggregation in the Hilbert space. The proposed UaGGP extends GGP by conducting feature aggregation in both the original input Euclidean space and the Hilbert space. In this section, we conduct ablation studies on the different aggregation manners. Respectively aggregating features in the original input Euclidean space, the Hilbert space and both spaces for GGP and UaGGP. The results are shown in Table 3.

Note that similar settings in previous section are used. Specifically, $n_L = 100$ labeled examples are used for *Pubmed* to ensure enough labels. Thus on *Pubmed*, the results for GGP in the Hilbert space and UaGGP in the Hi+Eu space are different from that in Table 2.

It can be seen from Table 3 that for both GGP and UaGGP, conducting aggregation in both spaces performs consistently better. Note that we aggregate features for one node with its one-hop neighbours. While in GCN(Kipf and Welling 2017), stacking multiple layers can take into account multi-hop neighbours. The UaGGP can also aggregate farther nodes by random-walk matrix like the LP(Zhu, Ghahramani, and Lafferty 2003), or deep Gaussian Processes(Salimbeni and Deisenroth 2017) as described in (Ng, Colombo, and Silva 2018).

Conclusion

In this paper, we propose a Gaussian Process based approach UaGGP for semi-supervised classification with graph structures. In UaGGP, the structural information of the graph is utilized in the feature aggregation and graph Laplacian regularization. By making use of the uncertainty of the Gaussian Processes model, we propose the graph Laplacian regularization based on the GP inferences and the symmetric Mahalanobis distance to deal with the structure uncertainty issues. Experimental comparison with multiple state-of-the-art methods on multiple benchmark datasets validated the effectiveness of the proposed approach. In the future, we plan to test the approach on more large datasets with a parallel implementation. Also it would be an interesting problem to extend the approach to more complicated tasks, such as when the examples are assigned with multiple labels or noisy labels.

References

Belkin, M.; Niyogi, P.; and Sindhwani. 2006. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research* 2399–2434.

Cai, H.; Zheng, V. W.; and Chang, K. C.-C. 2018. A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERIN* 30.

Chen, P., and Wei, D. 2018. On the supermodularity of active graph-based semi-supervised learning with stieltjes matrix regularization. *arXiv preprint arXiv:1804.03273*.

Chen, S.; Varma, R.; Sandryhaila, A.; and Kova**v**cević. 2015. Discrete signal processing on graphs: Sampling theory. *IEEE Transaction on Signal Processing* 63.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*.

Gadde, A., and Ortega, A. 2015. A probabilistic interpretation of sampling theory of graph signals. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 3257–3261.

Gadde, A.; Anis, A.; and Ortega, A. 2014. Active semi-supervised learning using sampling theory for graph signals. In *Proceedings of the 20th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Hensman, J.; Matthews, A. G. d. G.; and Ghahramani, Z. 2015. Scalable variational gaussian process classi

cation. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*.

Kingma, D. P., and Ba, J. L. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference Learning Representations*.

Kipf, T. N., and Welling, M. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th International Conference Learning Representations*.

Liao, R.; Brockschmidt, M.; Tarlow, D.; Gaunt, A. L.; Urtasun, R.; and Zemel, R. S. 2018. Graph partition neural networks for semi-supervised classification. *arXiv preprint arXiv:1803.06272*.

Ng, Y. C.; Colombo, N.; and Silva, R. 2018. Bayesian semi-supervised learning with graph gaussian processes. In *Advances in Neural Information Processing Systems*.

Orbach, M., and Crammer, K. 2012. Graph-based transduction with confidence. In *ECML/PKDD*, 323–338.

Perozzi, B.; AI-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*.

Rasmussen, C. E., and Williams, C. 2006. *Gaussian Processes for Machine Learning*. The MIT Press.

Salimbeni, H., and Deisenroth, M. P. 2017. Doubly stochastic variational inference for deep gaussian processes. In *Advances in Neural Information Processing Systems*.

Snelson, E., and Ghahramani, Z. 2005. Sparse gaussian processes using pseudo-inputs. In *Proceedings of the 18th International Conference on Neural Information Processing Systems*, 1257–1264.

Titsias, M. K. 2009. Variational learning of inducing variables in sparse gaussian processes. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*.

Vashishth, S.; Yadav, P.; Bhandari, M.; and Talukdar, P. 2019. Confidence-based graph convolutional networks for semi-supervised learning. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*.

Weston, J.; Ratle, F.; and Collobert, R. 2008. Deep learning via semi-supervised embedding. In *Proceedings of the 25th International Conference on Machine Learning*.

Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Philip, S. 2019. A comprehensive survey on graph neural networks. *arXiv preprint arXiv:1901.00596*.

Wu, T.-F.; Lin, C.-J.; and Weng, R. C. 2004. Probability estimates for multi-class classification by pairwise coupling. *Journal of Machine Learning Research* 975–1005.

Yang, Z.; Cohen, W. W.; and Salakhutdinow, R. 2016. Revisiting semi-supervised learning with graph embeddings. In *Proceedings of the 33rd International Conference on Machine Learning*.

Zhang, Y.; Zhang, Y.; Yeung, D.; Liu, C.; and Hou, X. 2010. Transductive learning on adaptive graphs. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*.

Zhang, Y.; Pal, S.; Coates, M.; and Üstebay, D. 2019. Bayesian graph convolutional neural networks for semi-supervised classification. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*.

Zhang, Z.; Cui, P.; and Zhu, W. 2018. Deep learning on graphs: A survey. *arXiv preprint arXiv:1812.04202*.

Zhou, J.; Cui, G.; Zhang, Z.; Yang, C. Y.; Liu, Z.; Wang, L.; Li, C.; and Sun, M. 2019. Graph neural networks: A review of methods and applications. *arXiv preprint arXiv:1812.08434*.

Zhu, X.; Ghahramani, Z.; and Lafferty, J. 2003. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the Twentieth International Conference on Machine Learning*.

Zhu, X. J. 2005. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences.