2023 IEEE 9th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP) | 979-8-3503-4452-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/CAMSAP58249.2023.10403487

Bayesian Self-Supervised Learning Using Local and Global Graph Information

Konstantinos D. Polyzos, Alireza Sadeghi, Georgios B. Giannakis

***Abstract*—Graph-guided learning has well-documented impact in a gamut of network science applications. A prototypical graph-guided learning task deals with semi-supervised learning over graphs, where the goal is to predict the nodal values or labels of unobserved nodes, by leveraging a few nodal observations along with the underlying graph structure. This is particularly challenging under privacy constraints or generally when acquiring nodal observations incurs high cost. In this context, the present work puts forth a Bayesian graph-driven self-supervised learning (Self-SL) approach that: (i) learns powerful nodal embeddings emanating from easier to solve auxiliary tasks that map local to global connectivity information; and, (ii) adopts an ensemble of Gaussian processes (EGPs) with adaptive weights as nodal embeddings are processed online. Unlike most existing deterministic approaches, the novel approach offers accurate estimates of the unobserved nodal values along with uncertainty quantification that is important especially in safety critical applications. Numerical tests on synthetic and real graph datasets showcase merits of the novel EGP-based Self-SL method.**

I. Introduction

Semi-supervised learning (Semi-SL) over graphs has gained popularity in recent years thanks to its impact in a gamut of network science applications, including e.g., social, financial and biological sciences [3]. Given a few nodal observations, the goal of Semi-SL is to reconstruct the nodal values of unobserved nodes [27]. Semi-SL approaches over graphs rely on the premise that neighboring nodes have similar nodal values. Such similarities manifest nonparametric models using e.g., graph kernels [8], [24], [21], [11], low-rank parametric models [23] or Gauss-Markov random fields [27]. Graph neural network (GNN) models have also been advocated in several network domains; see e.g [6], [9], [25]. GNN-based approaches typically operate in a batch form, they have large storage requirements, and satisfactory performance calls for a large number of training data. These requirements translate to high-cost Semi-SL over large-scale graphs [3].

Featuring affordable storage, the online multi-kernel approach in [22] uses the one-hop connectivity vector of each node to process per-node information in a streaming fashion. Also accounting for local nodal connectivity, a Bayesian online Gaussian Process (GP) based method with quantifiable uncertainty has been developed in [14], [17], and [15]. However, the *local connectivity information* leveraged in these works can have limited representation power for graph-guided inference, and can require considerably many nodal observations for training [3]. In addition, accounting for local information such as one-hop connectivity, further discourages application to large-scale graphs because the dimensionality of input grows linearly with the network size. To cope with the former, several methods advocate graph-guided active learning to judiciously select only a few most informative nodes to label [16], [13]. Nonetheless, active learning still necessitates extra labeling efforts, which can be challenging in practice.

To alleviate extra labeling in large-scale graphs but also allow for lower yet sufficient dimensionality of the input feature vector, one can rely on the paradigm of *self-supervised learning* (Self-SL) over graphs. Self-SL leverages *unlabeled* data to learn low-dimensional yet informative embedding representations per node. These embeddings are learned using 'pseudo-labels' obtained only from input features themselves; that is, from the graph structure and possibly nodal features if available [10], [4].

Self-SL approaches over graphs rely on GNNs to learn per-node local embeddings using observed local attributes, such as the per-node degree or local clustering coefficient and masked edges between nodes [5], or masked nodal features [26]. These approaches however, utilize as inputs only the global connectivity information captured by the adjacency matrix, which comes with high storage demands especially for large-scale graphs. Notwithstanding, they rely on additional nodal features to yield the embeddings. Learning memory-efficient node embeddings that capture local *and* global information with no need for extra nodal features, is still unexplored.

Contributions. To bypass extra labeling efforts and also allow for computationally-efficient learning methods due to low-dimensional yet informative input features, the present work develops a novel Self-SL approach that relies on both *local* and *global* connectivity features to learn nodal embeddings. The learned embeddings can be used for a wide range of Semi-SL over graph problems. In this contribution, they serve as input for a graph-guided Semi-SL regression task, which is carried out using a Bayesian online learning scheme that leverages an ensemble of (E)GPs. The goal is to identify an unknown function by adaptively learning the GP model weights on-the-fly as nodes are processed online, thus accommodating time-sensitive applications with reduced complexity and storage demands. Unlike existing approaches, the novel Bayesian Self-SL offers quantifiable uncertainty, and relies only on the graph structure without requiring extra nodal features or annotations. If extra nodal features are available, they can be leveraged to enrich nodal embeddings.

This work was supported by NSF grants 2312547, 2220292, 2212318, 2126052, 2103256, 2102312, and 2128593. The work of K. D. Polyzos was also supported by the Onassis Foundation Scholarship. Authors are with the Department of Electrical and Computer Engineering, University of Minnesota, USA. Emails: {*polyz003, sadeghi, georgios*}*@umn.edu.*

979-8-3503-4452-3/23/$31.00 ©2023 IEEE

Authorized licensed use limited to: University of Minnesota. Downloaded on March 06,2024 at 00:45:02 UTC from IEEE Xplore. Restrictions apply.

II. Problem formulation and Preliminaries

Consider a graph $\mathcal{G}$ comprising N nodes that form the vertex set $\mathcal{V} := \{1, \ldots, N\}$, and E edges that connect pairs of nodes. The connectivity among nodes is captured by the $N \times N$ adjacency matrix $\mathbf{A}$ whose (i,j)-th entry $a_{ij} := \mathbf{A}(i,j)$ is nonzero if node i is connected to node j. Let $f(\cdot) : \mathcal{V} \to \mathbb{R}$ be a real-valued function on this graph, which maps node $n \in \mathcal{V}$ to its corresponding ground-truth nodal value $f_n \in \mathbb{R}$ that is observed in additive noise ε_n as $y_n = f_n + \varepsilon_n$. For the Semi-SL task over graphs, only a few nodal observations $\{y_n, n \in \mathcal{O}\}$ are available, where the set $\mathcal{O}$ collects indices of *observed* nodes. Given observations $\{y_n, n \in \mathcal{O}\}$, the objective of Semi-SL over a graph is to estimate function values over a set $\mathcal{U}$ of *unobserved* nodes $\{\hat{y}_n, n \in \mathcal{U}\}$, where set $\mathcal{U}$ contains indices of such nodes.

To bypass computational complexity, improve privacy, and enhance generalization performance of semi-SL algorithms over graphs, recent contributions have advocated using only the one-hop connectivity vector $\mathbf{a}_n := \mathbf{A}(:, n)$ of node n as input feature vector in order to learn f online, where $f_n := f(\mathbf{a}_n)$ [14], [17], [15], [22], [16].

A. Learning with a single GP

When learning with Gaussian processes (GPs), f is viewed as random with a prior denoted by $f \sim \mathcal{GP}(0, \kappa(\mathbf{a}, \mathbf{a}'))$, where $\kappa(\mathbf{a}, \mathbf{a}')$ is the kernel function measuring the pairwise similarity between the connectivity vectors $\mathbf{a}$ and $\mathbf{a}'$. This implies that the $n \times 1$ vector of function evaluations $\mathbf{f}_n := [f(\mathbf{a}_1), \ldots, f(\mathbf{a}_n)]^\top$ ($^\top$ denotes transposition) with input matrix $\mathbf{A}_n := [\mathbf{a}_1, \ldots, \mathbf{a}_n]$, is multivariate Gaussian distributed $\forall n$; that is, $p(\mathbf{f}_n; \mathbf{A}_n) = \mathcal{N}(\mathbf{f}_n; \mathbf{0}_n, \mathbf{K}_n)$, where $\mathbf{K}_n$ is the kernel (covariance) matrix whose (m, m')-th entry is $[\mathbf{K}_n]_{m,m'} = \text{cov}(f(\mathbf{a}_m), f(\mathbf{a}_{m'})) := \kappa(\mathbf{a}_m, \mathbf{a}_{m'})$ [20].

Function evaluation vector $\mathbf{f}_n$ is related to nodal observations $\mathbf{y}_n := [y_1, \ldots, y_n]^\top$ through the batch conditional likelihood that upon assuming conditional independence across nodal measurements, it can be factored as $p(\mathbf{y}_n | \mathbf{f}_n; \mathbf{A}_n) = \prod_{n'=1}^{n} p(y_{n'} | f(\mathbf{a}_{n'}))$. In the regression task with $y_n = f(\mathbf{a}_n) + \varepsilon_n$ and $\varepsilon_n \sim \mathcal{N}(\varepsilon_n; 0, \sigma_n^2)$ uncorrelated across nodes, the conditional likelihood can be written as $p(\mathbf{y}_n | \mathbf{f}_n; \mathbf{A}_n) = \prod_{n'=1}^{n} \mathcal{N}(y_{n'}; f(\mathbf{a}_{n'}), \sigma_n^2)$. With the prior $p(\mathbf{f}_n; \mathbf{A}_n)$ and the likelihood $p(\mathbf{y}_n | \mathbf{f}_n; \mathbf{A}_n)$ at hand, it can be shown that the predictive probability density function (pdf) of the nodal value y_{n+1} corresponding to the unobserved node $n+1$ is Gaussian distributed [20], [17]; that is

$$p(y_{n+1} | \mathbf{y}_n; \mathbf{A}_n, \mathbf{a}_{n+1}) = \mathcal{N}(y_{n+1}; \hat{y}_{n+1|n}, \sigma_{n+1|n}^2) \tag{1}$$

with predictive mean and variance given by

$$\hat{y}_{n+1|n} = \mathbf{k}_{n+1}^\top (\mathbf{K}_n + \sigma_n^2 \mathbf{I}_n)^{-1} \mathbf{y}_n \tag{2a}$$

$$\sigma_{n+1|n}^2 = \kappa(\mathbf{a}_{n+1}, \mathbf{a}_{n+1}) - \mathbf{k}_{n+1}^\top (\mathbf{K}_n + \sigma_n^2 \mathbf{I}_n)^{-1} \mathbf{k}_{n+1} + \sigma_n^2 \tag{2b}$$

where $\mathbf{k}_{n+1} := [\kappa(\mathbf{a}_1, \mathbf{a}_{n+1}), \ldots, \kappa(\mathbf{a}_n, \mathbf{a}_{n+1})]^\top$. The mean in (2a) is an estimate for y_{n+1}, while the variance in (2b) quantifies the uncertainty of this estimate.

The single GP-based *batch* approach in (2) requires storage $\mathcal{O}(n^2)$, and complexity $\mathcal{O}(n^3)$, which can be prohibitive in large-scale network with large n. In addition, estimator accuracy depends on the pre-selected kernel κ, which thus regulates expressiveness of the sought function. The ensuing section shows how to bypass these limitations using the so-called random spectral features (RFs) based approximation.

B. Approximating a single GP with RFs

RF approximation begins with a *shift-invariant* standardized kernel $\bar{\kappa}(\mathbf{a}, \mathbf{a}') = \bar{\kappa}(\mathbf{a} - \mathbf{a}') = (1/\sigma_\theta^2)\kappa(\mathbf{a} - \mathbf{a}')$ satisfying

$$\bar{\kappa}(\mathbf{a} - \mathbf{a}') = \int \pi_{\bar{\kappa}}(\boldsymbol{\zeta}) e^{j\boldsymbol{\zeta}^\top (\mathbf{a} - \mathbf{a}')} d\boldsymbol{\zeta} = \mathbb{E}_{\pi_{\bar{\kappa}}} \left[e^{j\boldsymbol{\zeta}^\top (\mathbf{a} - \mathbf{a}')} \right]$$

with the power spectral density $\pi_{\bar{\kappa}}(\boldsymbol{\zeta})$ integrating to 1, and thus qualifying to be a pdf. For a *real*-valued $\bar{\kappa}$, it holds that $\bar{\kappa}(\mathbf{a} - \mathbf{a}') = \mathbb{E}_{\pi_{\bar{\kappa}}} \left[\cos(\boldsymbol{\zeta}^\top (\mathbf{a} - \mathbf{a}')) \right]$. Drawing sufficiently many i.i.d. deviates $\{\boldsymbol{\zeta}_i\}_{i=1}^{D}$ from $\pi_{\bar{\kappa}}(\boldsymbol{\zeta})$, $\bar{\kappa}$ can be approximated as $\bar{\kappa} \approx \check{\bar{\kappa}}(\mathbf{a}, \mathbf{a}') := D^{-1} \sum_{i=1}^{D} \cos\left(\boldsymbol{\zeta}_i^\top (\mathbf{a} - \mathbf{a}') \right)$. Upon defining the $2D \times 1$ RF vector

$$\boldsymbol{\phi}_{\boldsymbol{\zeta}}(\mathbf{a}) := \frac{1}{\sqrt{D}} \left[\sin(\boldsymbol{\zeta}_1^\top \mathbf{a}), \cos(\boldsymbol{\zeta}_1^\top \mathbf{a}) \cdots \sin(\boldsymbol{\zeta}_D^\top \mathbf{a}), \cos(\boldsymbol{\zeta}_D^\top \mathbf{a}) \right]^\top$$

the kernel approximant $\check{\bar{\kappa}}$ can be written as $\check{\bar{\kappa}}(\mathbf{a}, \mathbf{a}') := \boldsymbol{\phi}_{\boldsymbol{\zeta}}^\top(\mathbf{a}) \boldsymbol{\phi}_{\boldsymbol{\zeta}}(\mathbf{a}')$, which yields a *linear* and *parametric* approximant of the sought function, namely

$$\check{f}(\mathbf{a}) = \boldsymbol{\theta}^\top \boldsymbol{\phi}_{\boldsymbol{\zeta}}(\mathbf{a}), \quad \boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\theta}; \mathbf{0}_{2D}, \sigma_\theta^2 \mathbf{I}_{2D}) \tag{3}$$

leading to a GP prior $p(\check{\mathbf{f}}_n; \mathbf{A}_n) = \mathcal{N}(\check{\mathbf{f}}_n; \mathbf{0}_n, \sigma_\theta^2 \boldsymbol{\Phi}_n \boldsymbol{\Phi}_n^\top)$ with $\boldsymbol{\Phi}_n := \left[\boldsymbol{\phi}_{\boldsymbol{\zeta}}(\mathbf{a}_1), \ldots, \boldsymbol{\phi}_{\boldsymbol{\zeta}}(\mathbf{a}_n) \right]^\top$. Note that for $n > 2D$, the matrix $\sigma_\theta^2 \boldsymbol{\Phi}_n \boldsymbol{\Phi}_n^\top$ is a low-rank aproximant of $\mathbf{K}_n$, and can thus afford reduced complexity $\mathcal{O}(n(2D)^2)$ in (2) [17]. As in recursive Bayes, the parametric model (3) is amendable to *online* updates of the posterior $p(\boldsymbol{\theta} | \mathbf{y}_n; \mathbf{A}_n) = \mathcal{N}(\boldsymbol{\theta}; \hat{\boldsymbol{\theta}}_n, \boldsymbol{\Sigma}_n)$ per node n, thus alleviating the need for large storage [17].

III. Learning with an ensemble of GPs

Targeting a more expressive function model compared to that of a single GP with a pre-selected kernel, an ensemble (E) of M GP learners is advocated to estimate the sought function, where each learner $m \in \mathcal{M} := \{1, \ldots, M\}$ employs a distinct kernel selected from a set of available diverse kernels $\mathcal{K} := \{\kappa^m\}_{m=1}^{M}$. Considering a unique prior $p(\mathbf{f}_n | m; \mathbf{A}_n) = \mathcal{N}(\mathbf{f}_n; \mathbf{0}_n, \mathbf{K}_n^m)$ on f per GP learner m, an ensemble (E) GP learner adopts a weighted combination of all GP learners corresponding to the Gaussian mixture pdf

$$f(\mathbf{a}) \sim \sum_{m=1}^{M} w_n^m \mathcal{N}(\mathbf{f}_n; \mathbf{0}_n, \mathbf{K}_n^m) \ , \quad \sum_{m=1}^{M} w_n^m = 1. \tag{4}$$

The per-learner weight w_n^m can be viewed as the probability of learner m to describe the ground-truth function. Finally, the predictive pdf of the unobserved node $n+1$, can be written as $p(y_{n+1} | \mathbf{A}_n, \mathbf{y}_n, \mathbf{a}_{n+1}) = \sum_{m=1}^{M} w_n^m p(y_{n+1} | m, \mathbf{A}_{n+1}, \mathbf{y}_n)$ [12], [17]. The latter incurs $\mathcal{O}(Mn^3)$ complexity that can be further reduced via the RF-approximation for each GP model as delineated next.

Authorized licensed use limited to: University of Minnesota. Downloaded on March 06,2024 at 00:45:02 UTC from IEEE Xplore. Restrictions apply.

A. RF-based online EGP learner

Let each GP learner $m \in \mathcal{M}$ rely on its standardized and shift-invariant kernel $\bar{\kappa}^m = \kappa^m/\sigma_{\theta^m}^2$ with $\kappa^m \in \mathcal{K}$, and draw i.i.d. vectors $\{\boldsymbol{\zeta}_i^m\}_{i=1}^D$ from the power spectral density $\pi_{\bar{\kappa}^m}(\boldsymbol{\zeta})$ of $\bar{\kappa}^m$ to construct the RF vector $\boldsymbol{\phi}_{\boldsymbol{\zeta}}^m(\mathbf{a})$. This corresponds to the parametric generative function model for learner m being

$$p(\check{f}(\mathbf{a})|i=m,\boldsymbol{\theta}^m) = \delta(\check{f}(\mathbf{a}) - \boldsymbol{\phi}_{\boldsymbol{\zeta}}^{m\top}(\mathbf{a})\boldsymbol{\theta}^m) \qquad (5a)$$

$$p(\boldsymbol{\theta}^m) = \mathcal{N}(\boldsymbol{\theta}^m;\mathbf{0}_{2D},\sigma_{\theta^m}^2\mathbf{I}_{2D}) \qquad (5b)$$

that yields the Gaussian likelihood also parameterized by $\boldsymbol{\theta}^m$ as $p(y_n|\boldsymbol{\theta}^m,\mathbf{a}) = \mathcal{N}(y_n;\boldsymbol{\phi}_{\boldsymbol{\zeta}}^{m\top}(\mathbf{a})\boldsymbol{\theta}^m,\sigma_n^2)$. The latter along with the prior in (5b) lead to the posterior $p(\boldsymbol{\theta}^m|m,\mathbf{y}_n;\mathbf{A}_n) = \mathcal{N}(\boldsymbol{\theta}^m;\hat{\boldsymbol{\theta}}_n^m,\boldsymbol{\Sigma}_n^m)$ that enables prediction of unobserved nodal values by learner m. Next, we will show how $\{\hat{\boldsymbol{\theta}}_n^m,\boldsymbol{\Sigma}_n^m,w_n^m\}_m$ will be updated in a data-adaptive manner as nodes are processed online.

RF-based EGP prediction. Each GP learner m leverages its posterior $p(\boldsymbol{\theta}^m|m,\mathbf{y}_n;\mathbf{A}_n)$ to predict the pdf of y_{n+1} as

$$\begin{aligned}p(y_{n+1}|m,\mathbf{y}_n;\mathbf{A}_n,\mathbf{a}_{n+1})\\ &= \int p(y_{n+1}|m,\boldsymbol{\theta}^m;\mathbf{a}_{n+1})p(\boldsymbol{\theta}^m|m,\mathbf{y}_n;\mathbf{A}_n)d\boldsymbol{\theta}^m\\ &= \mathcal{N}(y_{n+1};\hat{y}_{n+1|n}^m,(\sigma_{n+1|n}^m)^2) \qquad (6)\end{aligned}$$

with mean and variance given by

$$\hat{y}_{n+1|n}^m = \boldsymbol{\phi}_{\boldsymbol{\zeta}}^{m\top}(\mathbf{a}_{n+1})\hat{\boldsymbol{\theta}}_n^m \qquad (7a)$$

$$(\sigma_{n+1|n}^m)^2 = \boldsymbol{\phi}_{\boldsymbol{\zeta}}^{m\top}(\mathbf{a}_{n+1})\boldsymbol{\Sigma}_n^m\boldsymbol{\phi}_{\boldsymbol{\zeta}}^m(\mathbf{a}_{n+1}) + \sigma_n^2\ . \qquad (7b)$$

The EGP learner combines the predictive pdfs of all M learners via the Gaussian mixture

$$p(y_{n+1}|\mathbf{y}_n;\mathbf{A}_n,\mathbf{a}_{n+1}) = \sum_{m=1}^M w_n^m\mathcal{N}(y_{n+1};\hat{y}_{n+1|n}^m,(\sigma_{n+1|n}^m)^2).$$

Then, the minimum mean-square error (MMSE) estimator of y_{n+1} along with the corresponding variance are given by

$$\hat{y}_{n+1|n} = \sum_{m=1}^M w_n^m\hat{y}_{n+1|n}^m \qquad (8a)$$

$$\sigma_{n+1|n}^2 = \sum_{m=1}^M w_n^m[(\sigma_{n+1|n}^m)^2 + (\hat{y}_{n+1|n} - \hat{y}_{n+1|n}^m)^2]\ . \qquad (8b)$$

where "$n+1|n$" indicates that only the nodal observation of node n and the model parameters after processing node n are used to predict y_{n+1}.

RF-based EGP correction. When y_{n+1} becomes available, each learner m leverages Bayes' rule to update its weight w_n^m and propagate its posterior pdf as

$$\begin{aligned}w_{n+1}^m &= \Pr(m|\mathbf{y}_{n+1};\mathbf{A}_{n+1}) = \frac{w_n^m p(y_{n+1}|m,\mathbf{y}_n;\mathbf{A}_{n+1})}{p(y_{n+1}|\mathbf{y}_n;\mathbf{A}_{n+1})}\\ &= \frac{w_n^m\mathcal{N}(y_{n+1};\hat{y}_{n+1|n}^m,(\sigma_{n+1|n}^m)^2)}{\sum_{m'=1}^M w_n^{m'}\mathcal{N}(y_{n+1};\hat{y}_{n+1|n}^{m'},(\sigma_{n+1|n}^{m'})^2)}, \qquad (9)\end{aligned}$$

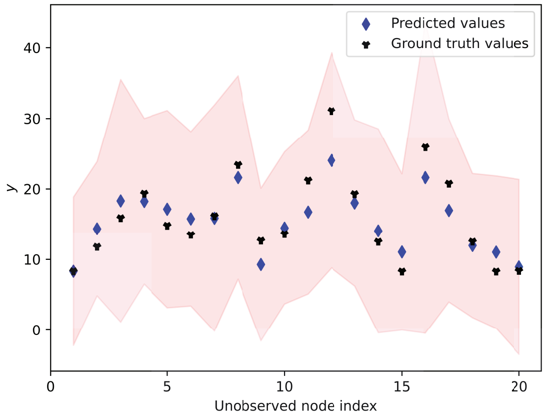

Fig. 1: Performance visualization across 20 unobserved nodes.

$$\begin{aligned}p(\boldsymbol{\theta}^m|\mathbf{y}_{n+1};\mathbf{A}_{n+1}) &= \frac{p(\boldsymbol{\theta}^m|\mathbf{y}_n;\mathbf{A}_n)p(y_{n+1}|\boldsymbol{\theta}^m;\mathbf{a}_{n+1})}{p(y_{n+1}|\mathbf{y}_n;\mathbf{A}_{n+1})}\\ &= \mathcal{N}(\boldsymbol{\theta}^m;\hat{\boldsymbol{\theta}}_{n+1}^m,\boldsymbol{\Sigma}_{n+1}^m) \qquad (10)\end{aligned}$$

where

$$\hat{\boldsymbol{\theta}}_{n+1}^m = \hat{\boldsymbol{\theta}}_n^m + (\sigma_{n+1|n}^m)^{-2}\boldsymbol{\Sigma}_n^m\boldsymbol{\phi}_{\boldsymbol{\zeta}}^m(\mathbf{a}_{n+1})(y_{n+1} - \hat{y}_{n+1|n}^m)$$

$$\boldsymbol{\Sigma}_{n+1}^m = \boldsymbol{\Sigma}_n^m - (\sigma_{n+1|n}^m)^{-2}\boldsymbol{\Sigma}_n^m\boldsymbol{\phi}_{\boldsymbol{\zeta}}^m(\mathbf{a}_{n+1})\boldsymbol{\phi}_{\boldsymbol{\zeta}}^{m\top}(\mathbf{a}_{n+1})\boldsymbol{\Sigma}_n^m\ .$$

Albeit offering a rich function space with scalability (the incurred per-iteration complexity is $\mathcal{O}(M((2D)^2 + 2DN))$), the developed EGP-based method solely relies on the one-hop connectivity vector $\mathbf{a}_n$, which may provide limited information about node n. In certain Semi-SL-related settings $\dim(\mathbf{a}_n) = N \gg |\mathcal{O}|$, which challenges learning the underlying function. To overcome these roadblocks, we develop a Self-SL method that aims at learning *low-dimensional* and *informative* embeddings per node n, that can be coupled with the EGP model as outlined next.

IV. Self-supervised learning with EGPs

In the context of semi-SL over graphs, our novel self-SL approach aims to learn rich nodal embeddings that capitalize on *local* and *global* connectivity features. Given graph $\mathcal{G}$, our self-SL algorithm relies on a neural network (NN) to learn a parametric *embedding function* $\boldsymbol{r}_{\boldsymbol{\vartheta}}(\mathbf{a}_n) : \mathcal{V} \to \mathbb{R}^d$, where $d \ll N$, and $\boldsymbol{\vartheta}$ collects the NN parameters. Scalability, memory savings, and computational efficiency considerations, motivate low-dimensional embedding with d chosen much smaller than the number of nodes. Capitalizing on $\boldsymbol{r}_{\boldsymbol{\vartheta}}(\cdot)$, the low-dimensional vector embedding per node n is obtained as $\boldsymbol{\rho}_n = \boldsymbol{r}_{\boldsymbol{\vartheta}^*}(\mathbf{a}_n)$, $n \in \mathcal{V}$, with the learned $\boldsymbol{\vartheta}^*$ obtained as

$$(\boldsymbol{\vartheta}^*, g^*) := \arg\min_{\boldsymbol{\vartheta},g}\sum_{n=1}^N \mathcal{L}\left(c_n, g\left(\boldsymbol{r}_{\boldsymbol{\vartheta}}(\mathbf{a}_n)\right)\right) \qquad (12)$$

where $\mathcal{L}(\cdot,\cdot) : \mathbb{R}\times\mathbb{R} \to \mathbb{R}$ is a loss function, c_n represents node n's pseudo-label obtained only by using the graph structure, and $g(\cdot) : \mathbb{R}^d \to \mathbb{R}$ is a (learnable) projection function that maps each embedding to the pseudo label pertinent to that node. The representation learning in (12) provides means to obtain embeddings $\{\boldsymbol{\rho}_n\}_{n=1}^N$ that encode *local* and *global*

Authorized licensed use limited to: University of Minnesota. Downloaded on March 06,2024 at 00:45:02 UTC from IEEE Xplore. Restrictions apply.

Table 1	**Synthetic SBM**		**Network delays**		**Temperature stations**	
Method	NMSE	NPLL	NMSE	NPLL	NMSE	NPLL
GradEGP (RBFs)	0.01936 ± 0.00025	-12.81 ± 1.20	0.116 ± 0.013	96.18 ± 1.70	0.1081 ± 0.0004	176.71 ± 8.00
GradEGP-feat (RBFs)	0.01935 ± 0.00045	-12.12 ± 1.39	0.106 ± 0.016	94.89 ± 2.03	0.1080 ± 0.0002	176.96 ± 9.20
SelfGradEGP (RBFs)	$\mathbf{0.01875 \pm 0.00029}$	-11.73 ± 2.70	0.060 ± 0.005	75.45 ± 1.50	0.1064 ± 0.0006	$\mathbf{146.44 \pm 4.39}$
GradEGP (mixed)	0.01958 ± 0.00041	-16.55 ± 1.50	0.132 ± 0.014	89.11 ± 2.05	0.0950 ± 0.0024	183.75 ± 4.35
GradEGP-feat (mixed)	0.01953 ± 0.00040	-16.94 ± 1.82	0.142 ± 0.023	89.76 ± 2.21	0.0939 ± 0.0024	176.06 ± 4.03
SelfGradEGP (mixed)	0.01934 ± 0.00070	-11.47 ± 1.82	$\mathbf{0.045 \pm 0.003}$	$\mathbf{69.72 \pm 1.58}$	$\mathbf{0.0865 \pm 0.0033}$	160.81 ± 6.79
GP	0.01950 ± 0.00061	-9.41 ± 2.81	0.114 ± 0.009	89.58 ± 2.34	0.1088 ± 0.0002	361.49 ± 0.03
GP-feat	0.01963 ± 0.00067	-16.77 ± 1.88	0.107 ± 0.021	87.86 ± 3.63	0.1088 ± 0.0003	361.47 ± 0.05
SelfGP	0.01901 ± 0.00037	$\mathbf{-18.07 \pm 1.48}$	0.055 ± 0.002	76.00 ± 1.50	0.1088 ± 0.0001	361.53 ± 0.02

connectivity per node. Input vector $\mathbf{a}_n$ captures local connectivity per node, while scalar output c_n denotes measurable global connectivity information.

Here, we adopt the square loss $\mathcal{L}(c_n, g(\boldsymbol{r}_{\boldsymbol{\vartheta}}(\mathbf{a}_n))) := (c_n - g(\boldsymbol{r}_{\boldsymbol{\vartheta}}(\mathbf{a}_n)))^2$, and the projection function can be thought to be an affine transformation; that is, $g(\boldsymbol{\rho}_n) := \mathbf{w}^\top \boldsymbol{\rho}_n + b$. The nested parametric function learning in (12) is solved iteratively using the back-propagation algorithm. To infuse global connectivity information in the learned embedding $\boldsymbol{\rho}_n$, we leverage the well appreciated eigenvector centrality of nodes as the pseudo-labels to be predicted, which measures the 'holistic influence' of a node in a network [7, pg. 90]. Nodes with high eigenvector centrality are more influential as they have many connections with other nodes. Let $\lambda_{\max}$ represent the largest eigenvalue of the adjacency matrix $\mathbf{A}$, with corresponding eigenvector $\boldsymbol{c}$, i.e., $\mathbf{A}\boldsymbol{c} = \lambda_{\max}\boldsymbol{c}$. The n-th entry c_n of $\boldsymbol{c}$ gives the centrality score of node n. Upon learning $\boldsymbol{\vartheta}^*$, we obtain node embeddings $\{\boldsymbol{\rho}_n := \boldsymbol{r}_{\boldsymbol{\vartheta}^*}(\mathbf{a}_n)\}_{n=1}^N$, which replace one-hop adjacency vectors $\{\mathbf{a}_n\}_{n=1}^N$ as EGP input in III.

V. Numerical tests

This section corroborates the performance of our proposed approach using both synthetic and real graph datasets.

Synthetic dataset. A synthetic graph consisting of $N = 60$ nodes is constructed using the stochastic block model comprising 10 communities; see e.g., [19]. The nodal value of node n is given by the n-th entry of the eigenvector corresponding to the *lowest* nonzero eigenvalue of the graph Laplacian $\mathbf{L} := \text{diag}(\mathbf{A}\mathbf{1}_N) - \mathbf{A}$ with $\mathbf{1}_N$ denoting an $N \times 1$ vector with all ones. The number of observed nodes is $|\mathcal{O}| = 10$ and the unobserved (test) ones is $|\mathcal{U}| = 50$.

Network delays dataset. A graph with $N = 70$ nodes is constructed, where nodes represent paths connecting two of 9 end-nodes on the Internet2backbone, and edges the shared links between any two paths [2]. The $\{y_n\}_{n=1}^N$ are the measured delays on these paths. The number of observed nodes is $|\mathcal{O}| = 15$, and of unobserved ones is $|\mathcal{U}| = 55$.

Temperature stations dataset. A graph with $N = 109$ nodes is constructed with nodes representing weather stations across the US, and edge weights the geographic distances between them [1]. Nodal values $\{y_n\}_{n=1}^N$ are the temperature measurements across the stations. Only $|\mathcal{O}| = 15$ measured temperatures are available and $|\mathcal{U}| = 94$ are to be predicted.

We compare our novel self-GP (SelfGP) and graph-adaptive EGP (SelfGradEGP) approaches against several benchmarks. We adopt the 'GradEGP' as one benchmark [17] which uses only the local features $\mathbf{a}_n$ as input per node n, while the 'GradEGP-feat' [20] uses $[\mathbf{a}_n, c_n]$, the single GP benchmark with input features $\mathbf{a}_n$ and the 'GP-feat' with $[\mathbf{a}_n, c_n]$. For the EGP-based approaches we adopt two distinct kernel dictionaries. The first one consists of 11 radial basis function (RBF) kernels with characteristic length scales $\{10^k\}_{k=-4}^6$ and the other comprises 4 kernels with distinct forms, namely RBF with and without automatic relevance determination [20], and Matern kernel with smoothness parameter $\nu = 3/2, 5/2$ [20]. For all RF-based approaches we set $D = 50$. To obtain the kernel parameters of GP-based approaches we maximize the marginal log-likelihood. For the Self-SL based approaches the embeddings are obtained using a feed-forward NN with only 2 layers. To train the NN parameters, we minimize the MSE loss using the Adam optimizer with learning rate 0.015 for 100 epochs. The learned embeddings have dimensionality $10, 15$, and 15 for the SBM, Network delay, and Temperature datasets.

The performance of all approaches is evaluated utilizing the normalized (N) MSE criterion to quantify the accuracy of predictions across unobserved nodes $n \in \mathcal{U}$, and the negative predictive log-likelihood (NPLL) to account for the associated *uncertainty* (cf. [18]). As corroborated by Table 1, the SelfGradEGP and SelfGP approaches outperform the alternatives in terms of NMSE. In addition, Table 1 illustrates that the novel approaches exhibit reduced uncertainty in most learning tasks, as evidenced by the smaller NPLL metric; see e.g., SelfGradEGP (with RBFs or mixed kernels) on the Network delay and Temperature datasets. This means besides accurate predictions, the proposed method quantifies well the uncertainty of these predictions via the predictive variance. Figure 1 depicts the predicted values of SelfGradEGP (mixed) on 20 randomly selected unobserved (test) nodes of the Network delay dataset, and the corresponing standard deviation σ-confidence intervals. It is observed that the ground truth nodal values fall within the the uncertainty intervals. All these observations demonstrate the importance of leveraging low-dimensional informative embeddings using local and global information obtained from the underlying graph.

VI. Conclusions

A novel Bayesian and graph-guided self-SL approach was introduced to solve semi-SL tasks over graphs. The proposed self-SL algorithm learns rich nodal embeddings leveraging both local and global connectivity information. The learned embeddings are then used as input features for the target graph-driven semi-SL task. The online Bayesian EGP employed offers accurate predictions of unobserved nodal values along with quantifiable uncertainty, low storage requirements, and reduced sample complexity.

Authorized licensed use limited to: University of Minnesota. Downloaded on March 06,2024 at 00:45:02 UTC from IEEE Xplore. Restrictions apply.

REFERENCES

[1] "1981-2010 U.S. climate normals," https://www.ncdc.noaa.gov/data-access/land-based-station-data/land-based-datasets/climate-normals/1981-2010-normals-data, [Online; accessed 29-April-2019].

[2] "One-way ping internet2," http://software.internet2.edu/owamp/, [Online; accessed 29-April-2019].

[3] G. B. Giannakis, Y. Shen, and G. V. Karanikolas, "Topology identification and learning over graphs: Accounting for nonlinearities and dynamics," *Proc. of the IEEE*, vol. 106, no. 5, pp. 787–807, May 2018.

[4] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A survey on contrastive self-supervised learning," *Technologies*, vol. 9, no. 1, p. 2, 2020.

[5] W. Jin, T. Derr, H. Liu, Y. Wang, S. Wang, Z. Liu, and J. Tang, "Self-supervised learning on graphs: Deep insights and new direction," *arXiv:2006.10141*, 2020.

[6] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *Proc. Int. Conf. Learn. Represent.*, Apr. 2017.

[7] E. D. Kolaczyk, *Statistical Analysis of Network Data*. Springer-Verlag New York, 2009.

[8] R. I. Kondor and J. Lafferty, "Diffusion kernels on graphs and other discrete structures," *Proc. Int. Conf. Mach. Learn.*, pp. 315–322, Jul. 2002.

[9] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Proc. of AAAI Conf. on Artificial Intel.*, New Orleans, Louisiana, Feb. 2018.

[10] Y. Liu, M. Jin, S. Pan, C. Zhou, Y. Zheng, F. Xia, and P. Yu, "Graph self-supervised learning: A survey," *IEEE Trans. Knowledge and Data Eng.*, 2022.

[11] Q. Lu, V. N. Ioannidis, and G. B. Giannakis, "Semi-supervised learning of processes over multi-relational graphs," in *Proc. IEEE Int. Conf. Acoust., Speech, Sig. Process.*, 2020, pp. 5560–5564.

[12] Q. Lu, G. Karanikolas, Y. Shen, and G. B. Giannakis, "Ensemble Gaussian processes with spectral features for online interactive learning with scalability," *Proc. Int. Conf. Artificial Intel. and Stats.*, June 2020.

[13] Y. C. Ng, N. Colombo, and R. Silva, "Bayesian semi-supervised learning with graph Gaussian processes," *Proc. Advances Neural Inf. Process. Syst.*, vol. 31, 2018.

[14] K. D. Polyzos, Q. Lu, and G. B. Giannakis, "Graph-adaptive incremental learning using an ensemble of Gaussian process experts," *Proc. IEEE Int. Conf. Acoust., Speech, Sig. Process.*, June 2021.

[15] ——, "Online graph-guided inference using ensemble Gaussian processes of egonet features," in *Proc. Asilomar Conf. Sig., Syst., Comput.*, 2021, pp. 182–186.

[16] ——, "Active sampling over graphs for Bayesian reconstruction with Gaussian ensembles," in *Proc. Asilomar Conf. Sig., Syst., Comput.*, 2022, pp. 58–64.

[17] ——, "Ensemble Gaussian processes for online learning over graphs with adaptivity and scalability," *IEEE Trans. Sig. Process.*, vol. 70, pp. 17–30, 2022.

[18] ——, "Weighted ensembles for active learning with adaptivity," *arXiv:2206.05009*, 2022.

[19] K. D. Polyzos, C. Mavromatis, V. N. Ioannidis, and G. B. Giannakis, "Unveiling anomalous edges and nominal connectivity of attributed networks," *Proc. Asilomar Conf. Sig., Syst., Comput.*, Nov. 2020.

[20] C. E. Rasmussen and C. K. Williams, *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006.

[21] D. Romero, M. Ma, and G. B. Giannakis, "Kernel-based reconstruction of graph signals," *IEEE Trans. Sig. Process.*, vol. 65, no. 3, pp. 764–778, 2017.

[22] Y. Shen, G. Leus, and G. B. Giannakis, "Online graph-adaptive learning with scalability and privacy," *IEEE Trans. Sig. Process.*, vol. 67, no. 9, May 2019.

[23] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Sig. Process. Mag.*, vol. 30, no. 3, pp. 83–98, May 2013.

[24] A. J. Smola and R. Kondor, "Kernels and regularization on graphs," in *Learning Theory and Kernel Machines*. Springer, 2003, pp. 144–158.

[25] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Trans. on Neural Net. and Learn. Syst.*, 2020.

[26] Y. You, T. Chen, Z. Wang, and Y. Shen, "When does self-supervision help graph convolutional networks?" in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 10 871–10 880.

[27] X. Zhu, Z. Ghahramani, and J. D. Lafferty, "Semi-supervised learning using Gaussian fields and harmonic functions," *Proc. Int. Conf. Mach. Learn.*, pp. 912–919, 2003.

Authorized licensed use limited to: University of Minnesota. Downloaded on March 06,2024 at 00:45:02 UTC from IEEE Xplore. Restrictions apply.