Learning Regularization for Graph Inverse Problems

Moshe Eliasof[1], Md Shahriar Rahim Siddiqui[2], Carola-Bibiane Schönlieb[1], Eldad Haber[3]

[1]Department of Applied Mathematics, University of Cambridge, Cambridge, United Kingdom
[2]Department of Physics and Astronomy, University of British Columbia, Vancouver, Canada
[3]Department of Earth, Ocean and Atmospheric Sciences, University of British Columbia, Vancouver, Canada
me532@cam.ac.uk, shahsiddiqui@phas.ubc.ca, cbs31@cam.ac.uk, ehaber@eoas.ubc.ca

Abstract

In recent years, Graph Neural Networks (GNNs) have been utilized for various applications ranging from drug discovery to network design and social networks. In many applications, it is impossible to observe some properties of the graph directly; instead, noisy and indirect measurements of these properties are available. These scenarios are coined as Graph Inverse Problems (GRIPs). In this work, we introduce a framework leveraging GNNs to solve GRIPs. The framework is based on a combination of likelihood and prior terms, which are used to find a solution that fits the data while adhering to learned prior information. Specifically, we propose to combine recent deep learning techniques that were developed for inverse problems, together with GNN architectures, to formulate and solve GRIPs. We study our approach on a number of representative problems that demonstrate the effectiveness of the framework.

1 Introduction

Graphs represent an elegant and powerful way to describe and analyze connected data, capturing the relationships and interactions between different entities in a structured manner. This mathematical framework allows for the exploration and exploitation of both the structure and the intrinsic properties of the data, enabling the discovery of hidden connections and dependencies that may not be evident by directly processing each data point. In contrast, by representing data as nodes and their relationships as edges, i.e., as graphs, we can facilitate a deeper understanding of complex systems, ranging from social networks and biological systems to communication networks, as well as financial markets.

In recent years, graph machine learning frameworks were developed, predominantly, Graph Neural Networks (Scarselli et al. 2008; Bronstein et al. 2021), which are able to model and learn complex patterns in graph data. These recent advancements make it possible to perform a wide array of applications, such as node classification (Kipf and Welling 2016; Defferrard, Bresson, and Vandergheynst 2016), community detection (Chen, Li, and Bruna 2019), drug discovery (Jiang et al. 2021), and solving combinatorial problems (Schuetz, Brubaker, and Katzgraber 2022; Eliasof and Haber 2024).

However, despite their high and vast utilization, in many practical scenarios, the direct observation of properties of the graph is not feasible or limited. For example, the knowledge of node labels might be partial. Instead, we often have access only to the outcomes of certain operations or processes that act upon the graph properties to be recovered. Additional examples are, in road network systems, where we might observe traffic patterns rather than the underlying network features, or, in social networks, we might see interaction patterns without having direct access to the strength of the network's relationships. This observation poses a significant research question: *how can we recover or infer the original graph properties from these observed outcomes?*

Further complexity typically arises from the fact that this question is often ill-posed. This implies that given the data, there is more than one answer to the question and the solutions can be unstable with respect to even small perturbations in the data.

To address this question and its challenges, we begin by defining Graph Inverse Problems (GRIPs), highlighting a key difference between classical inverse problems that are commonly solved on structured grids and those on graphs. This difference directly stems from the presence of the graph, which provides additional structure. Furthermore, in many GRIP cases, there exist additional features on the graphs (throughout this paper, we will refer to these features as *meta-data*) which may not be directly related to the observed data associated with the inverse problem, but can be leveraged to obtain better estimates for the solution of the inverse problem. Following the definition of such problems, and a number of examples, we discuss several viable approaches, from classical to neural methods, and evaluate them on several datasets and GRIPs.

Importantly, we note that, although different GRIPs share many characteristics, existing methods to solve them, such as the seminal works in Yan, Fang, and He (2023); Huang et al. (2023); Ling et al. (2024) are specialized for solving specific problems, as we discuss in Appendix A. In this work, we focus on extending these ideas to a general framework that can be applied to various GRIPs. Previous works on solving similar problems do not use any meta-data as additional features, and consider the likelihood (i.e., data fit) as a part of the network; Rather, they focus on applying various GNN architectures to solve the problem directly from

the observed data, while in this work, we show the effectiveness of including meta-data in GRIPs. Finally, as we discuss in Appendix A, most GNN-based methods known to us that have been proposed for the solution of GRIPs use the same graph structure for training and prediction, where the variable part is the observed data, given as node features. This choice, while useful in many cases, can be limiting for many practical problems, where not only the observed data changes but also the underlying graph. These three key concepts distinguish our proposed GRIP framework from existing works.

Main Contributions. This paper advances the bridge between the field of inverse problems and graph machine learning by an overarching methodology for the solution of inverse problems that reside on graphs, that is, GRIP. By integrating the principles of learned regularization techniques (Adler and Öktem 2017; Mardani et al. 2018), with the flexibility of GNNs (Khemani et al. 2024), this work aims to develop a unified framework that leverages the strengths of both approaches, to solve GRIP. We demonstrate our framework on several key GRIPs, such as graph feature completion, source estimation, inverse graph transport, and the nonlinear edge recovery problem. We note that the proposed framework not only enhances the existing techniques for GRIP but also broadens the scope of GNN applications.

2 Graph Inverse Problems

In this section, we define and provide background on inverse problems defined on graphs, followed by examples of several key inverse problems with real-world applications.

Notations. Throughout this paper, we consider input features, that reside on a graph $\mathcal{G}$ defined by its set of n nodes $\mathcal{V}$ and m edges $\mathcal{E}$. The features can be associated with either nodes or edges, or both, depending on the inverse problem.

In addition, we consider an observation that is a function of the recoverable properties, and the graph structure. To be more precise, the features are divided into *meta-data*, $\mathbf{f}_{\mathrm{M}} \in \mathcal{F}$, and states that describe the properties, denoted by $\mathbf{x} = [\mathbf{x}_N, \mathbf{x}_E] \in \mathcal{X}$. The state $\mathbf{x}_N$ is associated with the nodes and the state $\mathbf{x}_E$ is associated with the edge. We assume that we have observed data, i.e., measurements, $\mathbf{d}^{\mathrm{obs}} \in \mathcal{D}$, on the states $\mathbf{x}$. Note that, the difference between the meta-data $\mathbf{f}_{\mathrm{M}}$, the states, $\mathbf{x}$, and the observed data $\mathbf{d}^{\mathrm{obs}}$ is important; While $\mathbf{f}_{\mathrm{M}}$ and $\mathbf{x}$ reside on the nodes or edges of the graph, the observed data $\mathbf{d}^{\mathrm{obs}}$, in general, reside in a different space that does not share the same domain. An example is having observed data that is related to a global measurement from the graph, e.g., average node degree.

In the context of inverse problems, the observed data is the observation from which the desired state is to be recovered, while the (optional) meta-data serves as additional information that can be used in the inverse problem solution process. Namely, we assume the following connection between the observed data $\mathbf{d}^{\mathrm{obs}}$ and the state $\mathbf{x}$:

$$\mathbf{d}^{\mathrm{obs}} = F(\mathbf{x}; \mathcal{G}) + \boldsymbol{\varepsilon} \tag{1}$$

where $F : \mathcal{X} \to \mathcal{D}$ is the forward map of the problem that maps the states $\mathbf{x}$ that reside on the graph to the data space $\mathcal{D}$. The vector $\boldsymbol{\varepsilon}$ is some noise that, for simplicity, is assumed to be $\sigma \sim N(0, \sigma^2\mathbf{I})$. While for some problems, the noise can be substantial, for many practical problems (e.g., graph segmentation), the noise can be negligent.

In the *forward* problem, we are given the graph $\mathcal{G}$, the states $\mathbf{x}$, from which we can obtain the forward problem data, $F(\mathbf{x}, \mathcal{G})$. In the *inverse* problem, we are given possibly noisy observations, $\mathbf{d}^{\mathrm{obs}}$, the graph $\mathcal{G}$, and optionally meta-data, $\mathbf{f}_{\mathrm{M}}$. The goal is to estimate the states $\mathbf{x}$. In what follows, we define and discuss several key graph inverse problems.

2.1 Property Completion (Figure 1)

Consider the case where the forward problem reads

$$F(\mathbf{x}, \mathcal{G}) = \mathbf{I}_n^{\mathrm{P}}\mathbf{x}, \tag{2}$$

where $\mathbf{I}_n^{\mathrm{P}} \in \mathbb{R}^{p \times n}$ is a subset of p rows from an $n \times n$ identity matrix. In this case, we need to reconstruct the graph properties from the partial, potentially noisy data $\mathbf{d}^{\mathrm{obs}}$. This problem is commonly solved in the GNN literature (see e.g., (Bronstein et al. 2017)). We note that, in the absence of noise, i.e. $\epsilon = 0$, the inverse problem defined by the operator in Equation (2) coincides with the typical settings of semi-supervised node classification (Kipf and Welling 2016). An illustration of the problem is provided in Figure 1.

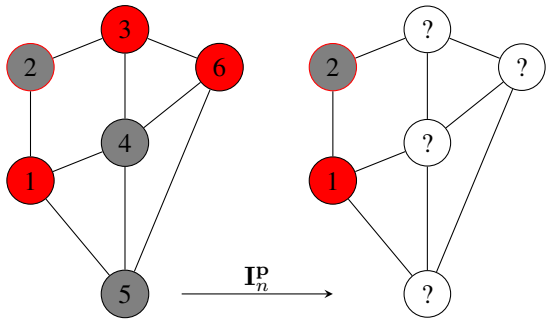

Figure 1: The forward problem of graph density completion.

2.2 Inverse Source Estimation (Figure 2)

A second popular problem is inverse source estimation, sometimes referred to as the source localization problem (Huang et al. 2023). This problem occurs by modeling the spread of information on a graph. Specifically, we consider the case where the forward problem reads

$$F(\mathbf{x}, \mathcal{G}) = \mathbf{P}^k\mathbf{x}^0, \tag{3}$$

where $\mathbf{P}$ is a Markov transition matrix that spreads the information from nodes to their neighbors. In the canonical setting, k represents some time frame where the information spreads from one node to its neighbors. If we observe the system after k time frames, we obtain (3). A popular instance of this problem is the case where $\mathbf{P}$ is the degree normalized adjacency matrix, i.e., $\mathbf{P} = \mathbf{D}^{-1}\mathbf{A}$, where $\mathbf{D}$ is the node degree matrix, and $\mathbf{A}$ is the binary adjacency matrix. The goal is then to find the source $\mathbf{x}^0$, as illustrated in Figure 2.

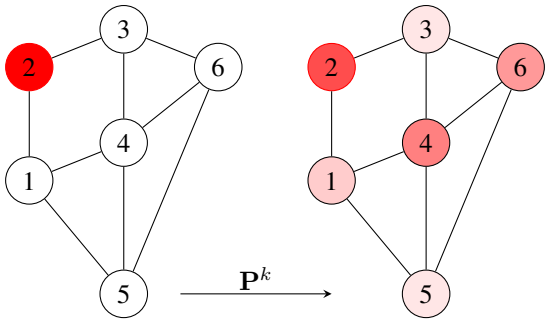

Figure 2: A source graph on the left is diffused over time, with transition matrix $\mathbf{P}^k$ obtaining a target graph on the right. The goal of the inverse problem is to identify the source given the target.

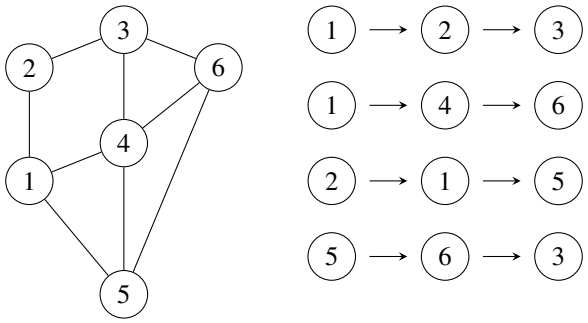

Figure 3: The graph transport problem. The four sampled paths of length 3 on the right are obtained from the graph on the left. The data is the sum of the node properties.

At this point, it is important to note that for the source estimation problem, the problem becomes harder as k becomes larger because an increase in k exponentially damps the information in $\mathbf{x}$ that is associated with eigenvalues that are smaller than 1 (Golub and Loan 1983).

2.3 Inverse Graph Transport (Figure 3)

We consider the inverse graph transport, a problem where the measurement is obtained through an averaging of the state along a walk on the graph. Specifically, we consider the case where the observed data can be expressed as

$$F(\mathbf{x}, \mathcal{G})_k = \sum_{j=0}^{L-1} \mathbf{x}_{\Gamma_k^j} \quad k = 1, \ldots, K \tag{4}$$

Here, we are given a set of K paths $\{\Gamma_k\}_{k=1}^K$, of length L, such that the path Γ_k is a vector of length L whose entries are the node indices in which the path traverses through. In Equation (4) we use the notation Γ_k^j to obtain the j-th node index on the path Γ_k, in order to sum over the node features that are in the path. We note that the graph inverse transport problem is similar to the ray tomography problem in a continuous setting (Natterer 2001). The goal is to recover the original state $\mathbf{x}$ given observations on their partial sums.

For this problem, it is important to note that the data does not have an obvious graph structure. In fact, the data can be much larger than the graph. Some nodes can be sampled multiple times and some very few or not at all.

Remark. The problems defined in Equations (2)–(4) are linear. Next, we show an example of a *nonlinear* problem, which is considered more complex (Manríquez et al. 2021).

2.4 Edge Property Recovery (Figure 4)

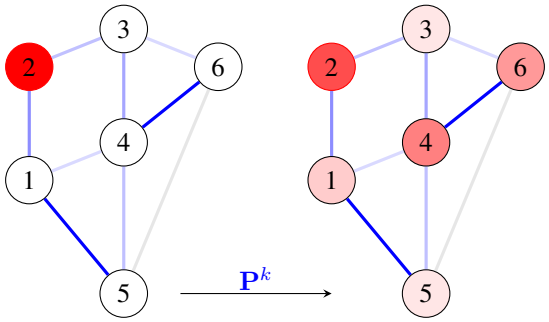

Figure 4: A **known** source graph on the left is diffused over time, with an **unknown** transition matrix $\mathbf{P}^k$ determined by the edge weights, obtaining the target graph on the right. The goal of the inverse problem is to identify the edge weights given the source and the target.

Consider the case of source estimation in Equation (3), but with **known** source and **unknown** edge properties where

$$\mathbf{x}_N^{(k+1)} = \mathbf{P}(\mathbf{x}_E)\mathbf{x}_N^{(k)} \quad k = 0, \ldots, K-1. \tag{5}$$

Here $\mathbf{P}(\mathbf{x}_E)$ is a Markov transition matrix that depends on the property of the edge. Assuming that we have the history $\mathbf{d}^{\text{obs}} = [\mathbf{x}_N^{(0)}, \ldots, \mathbf{x}_N^{(K)}] + \boldsymbol{\varepsilon}$, the goal is to find the edge property, that is, the edge states, $\mathbf{x}_E$ from the data.

Note that this problem is a nonlinear extension of the source estimation problem. Here, rather than assuming that the source $\mathbf{x}_N$ is the unknown state to be recovered, we have the matrix $\mathbf{P}$, which encodes edge properties (e.g., edge weights) as the unknown states to be recovered.

For the forward problems presented in this section, their inverse problems are assumed to be ill-posed. That is, the operator F in Equation (1) has a large, non-trivial null space or, such that the singular values of its Jacobian quickly decay to 0. This implies that a small perturbation in the data can result in a large perturbation in the estimated solution (see, e.g., (Hansen 1997)). Therefore, to obtain an estimate of the solution, some regularization is required. Such regularization can vary from classical techniques like Tikhonov regularization (Tikhonov 1950) (similar to weight-decay in neural networks), smoothness-based regularization (Nadler, Srebro, and Zhou 2009), total-variation (Rudin, Osher, and Fatemi 1992), as well as learnable neural regularizers (Adler and Öktem 2017). In previous works on graph inverse problem (Huang et al. 2023), such a regularization was achieved implicitly by learning a map, $F^\dagger$, from the data $\mathbf{d}^{\text{obs}}$ to the solution $\mathbf{x}$. Nonetheless, as has been shown for other, non-graph inverse problems (Adler and Öktem 2017; Eliasof, Haber, and Treister 2023), even when the data fit is a part of

the training, at inference, such a map can lead to reconstructions that do not fit the observed data – which can lead to non-feasible solutions. Mathematically, these findings mean that the residual between the predicted solution and the observed data, defined as:

$$\|F(F^{\dagger}(\mathbf{d}^{\text{obs}}) - \mathbf{d}^{\text{obs}}\|$$

may not be small and, in some cases, diverge. Therefore, in Section 3, we explore methodologies that blend graph-based regularization with the forward problem, yielding algorithms that adhere both to a priori information (that is, the prior learned in $F^{\dagger}$) and the given observed data $\mathbf{d}^{\text{obs}}$.

3 Methodologies for Inverse Problems

In this section, we develop the ideas presented in the paper. We start by reviewing the basic concepts that are used for the solution of inverse problems and then explain how they can be adopted and modified to work with the graph structure and how meta-data can be used. Later, in Section 4, we prescribe specific methods to solve GRIP.

Goal. The objective in solving Inverse Problems is to estimate a solution $\widehat{\mathbf{x}}$ that (i) fits the data in Equation (1) to some prescribed error, and (ii) adheres to a-priori information given by a regularizer. Two approaches to solving such problems are commonly used. The *first*, is based on a variational method and the *second* is to embed the solution with an (inverse) scale-space dynamical system. We now shortly review both techniques and their behavior.

Variational Techniques. Variational methods formulate inverse problems as an optimization task, seeking to minimize a functional that balances fidelity to the observed data with the imposition of prior knowledge or regularization (see, e.g., (Engl, Hanke, and Neubauer 1996; Calvetti and Somersalo 2005; Tenorio et al. 2011)). They have a strong relation to the Maximum Aposteriori Estimate (MAP) when considering a Bayesian approach (Tarantola 1987). In the context of inverse problems, the objective functional includes a data fitting term, which ensures that the solution is consistent with the observed data, and a regularization term, which incorporates prior assumptions about the solution, such as smoothness, sparsity, or other structural properties. Here, we consider regularization techniques where the estimated $\widehat{\mathbf{x}}$ is obtained by solving a regularized data fitting problem:

$$\widehat{\mathbf{z}} = \text{argmin}_{\mathbf{z}} \frac{1}{2}\|F(\mathbf{E}\mathbf{z}) - \mathbf{d}^{\text{obs}}\|^2 + \alpha \mathbf{R}(\mathbf{z}, \boldsymbol{\theta}), \quad (6a)$$

$$\widehat{\mathbf{x}} = \mathbf{E}\widehat{\mathbf{z}}. \quad (6b)$$

Here $\mathbf{E}$ is a learnable embedding function, and $\mathbf{R}(\mathbf{z}, \boldsymbol{\theta})$ is a regularization functional that depends on learnable parameters $\boldsymbol{\theta}$, and α is a non-negative coefficient that balances the terms in Equation (6a). The 'art' of obtaining meaningful solutions is transformed into the learning of an appropriate regularization functional $\mathbf{R}$ and an embedding $\mathbf{E}$.

Inverse Scale-Space Methods. One of the difficulties in solving the optimization proposed in Variational Methods, as in Equation (6a), is the appropriate choice of the relative weight α, between the data fitting and regularization terms (Nagy and Hansen 2006). Inverse scale-space methods (Burger et al. 2006) propose an alternative approach by casting the optimization problem to a dynamical system that starts with a coarse approximation of the solution and progressively incorporates finer details. By initially focusing on the low-frequency details and gradually refining the solution to encode high-frequency details, inverse scale-space techniques can achieve a balance between fidelity to the data and the incorporation of prior knowledge. The choice of the regularization parameter is replaced by the choice of the stopping time, which is often more straightforward.

In practice, an inverse scale-space approach is obtained by integrating an ordinary differential equation of the form

$$\frac{d\mathbf{z}}{dt} = \mathbf{E}(t)^{\top}\mathbf{J}(t)^{\top}\left(\mathbf{d}^{\text{obs}} - F(\mathbf{E}(t)\mathbf{z})\right) - s(\mathbf{z}, \boldsymbol{\theta}(t)), \quad (7)$$

from time 0 to time T, with $\mathbf{z}(0) = \mathbf{z}_0$, which is determined by the level of fitting the data. Here, $\mathbf{E}(t)$ is a time-dependent embedding, $\mathbf{J}(t)$ is the Jacobian of F with respect to its argument, and s is a function referred to as the score. That is, in order to use a scale space approach for the solution of the problem, one is required to choose time-dependent embedding and a score function, s. The flexibility of a time-dependent embedding $\mathbf{E}$ and regularization s was shown to overcome issues such as local minima (Eliasof, Haber, and Treister 2024). We note that, while Inverse scale-space methods offer more flexibility, they have less theoretical understanding compared with variational methods. Additionally, scale space methods can be reduced to solving the problem in Equation (6a) using gradient descent, if we choose $s = \nabla\mathbf{R}$ and $\mathbf{E}$ to be time-invariant; however, typically, in Scale Space methods, s and $\mathbf{E}$ are time-dependent.

4 A Framework for Solving GRIP

In Section 3, we discussed generic regularization techniques that have been used to solve general inverse problems. In this Section, we discuss how such regularization techniques can be adopted to solve GRIP. In particular, we identify two key differences between inverse problems on a *structured* grid, and inverse problems on *graphs*, as discussed below:

(i) Graph-Informed Regularizers. Because the problem is defined on a graph, it is possible and desirable to leverage the graph structure to obtain additional insights on the support of the problem. Regularization techniques can incorporate the connectivity and topology of the graph, enabling more informed and precise recovery of the property $\mathbf{x}$. For example, smoothness constraints can be imposed, effectively promoting solutions where neighboring nodes have similar values, which is particularly useful in applications like homophilic node classification (Zhou et al. 2004).

(ii) Utilization of Meta-Data in Graphs. While the observed data $\mathbf{d}^{\text{obs}}$, defined in Equation (1), depends solely on states $\mathbf{x}$, in many cases, each node or edge in the graph is associated with additional meta-data $\mathbf{f}_{\text{M}}$. This meta-data, which may include attributes such as node positions, edge weights, or additional features, is not required to compute $\mathbf{d}^{\text{obs}}$, but it provides valuable supplementary information that can enhance the regularization process. For example, in applications related to 3D Computer Vision, where point clouds and graphs are processed, it is common to have multiple types of features, such as xyz coordinates, point nor-

mals, node-wise labels, as well as a global label of the graph. Additionally, we note that in the absence of observed data $\mathbf{d}^{\text{obs}}$, the proposed framework reduces to standard GNN architectures, as discussed below. Overall, this two-fold consideration of the graph structure and meta-data enables a more holistic approach to solving GRIP, as we now discuss.

4.1 Classical Graph Regularization

We now discuss two classical regularization techniques that can be used for the solution of GRIP.

Laplacian Regularization. A classical regularizer that has been used extensively, especially in the case of graph completion (Nadler, Srebro, and Zhou 2009) is based on the graph Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{A}$, whose smoothness regularization is defined as

$$\mathbf{R}(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top \mathbf{L}\mathbf{x}. \tag{8}$$

To solve the optimization problem in Equation (6a), where $\mathbf{E} = \mathbf{I}$, it is common (Nadler, Srebro, and Zhou 2009) to use a scaled gradient descent method that reads:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{H}^{-1}\left(\mathbf{J}_k^\top (F(\mathbf{x}_k) - \mathbf{d}^{\text{obs}}) + \alpha \mathbf{L}\mathbf{x}_k\right). \tag{9}$$

Here, $\mathbf{H}$ is a symmetric positive-definite matrix, chosen to accelerate the convergence of the method, and $\mathbf{J}_k$ is the Jacobian of F at the k-th step. The choice $\mathbf{H}^{-1} = \mu\mathbf{I}$ yields the gradient descent. Alternatively, choosing $\mathbf{H}^{-1} = \mu\mathbf{L}^{-1},\ \mu > 0$ and $\alpha = 0$ we obtain the scale space iteration:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mu\mathbf{L}^\dagger \mathbf{J}_k^\top (F(\mathbf{x}_k) - \mathbf{d}^{\text{obs}}). \tag{10}$$

It was shown in Calvetti, Reichel, and Shuibi (2005) that early termination of the iteration in Equation (10) gives similar results to tuning the regularization parameter α in Equation (9). Thus, it is possible to use Equation (9) and tune α as a hyper-parameter, or Equation (10) and tune the number of iterations as a hyper-parameter and obtain similar results.

Tikhonov Regularization. The Tikhonov Regularization technique (Tikhonov and Arsenin 1977), which promotes solutions with low norms, is defined by replacing the Laplacian $\mathbf{L}$ in Equation (8) by the identity matrix $\mathbf{I}$.

4.2 Neural Graph Regularization

We discuss neural approaches for learning regularization for GRIPs. All methods include terms that are dependent on the graph $\mathcal{G}$, and are implemented using a GNN. We provide specific details on the GNN architecture in Appendix C.

Variational Methods. We consider the use of variational regularization, discussed in Section 3, where the idea is to *learn* the regularization using a GNN. In particular, when working with graph data, there often exist additional, meta-data, $\mathbf{f}_M$ for each node. Thus, we modify the regularization operator proposed in (Eliasof, Haber, and Treister 2023) to include meta-data, such that this regularization $\mathbf{R}(\mathbf{x}_1, \ldots, \mathbf{x}_L, \mathbf{f}_M, \{\boldsymbol{\theta}_l\}_{l=1}^L)$ reads:

$$\mathbf{R} = \sum_{l=1}^{L-1} \frac{1}{2}\|\mathbf{x}_{l+1} - \mathbf{x}_l\|^2 \ + \ \sum_{l=1}^{L-1} \mathbf{\Phi}(\mathbf{x}_l, \mathbf{f}_M, \mathcal{G}, \boldsymbol{\theta}_l). \tag{11}$$

Here, the solution $\mathbf{x} = \mathbf{x}_L$ is embedded in the hidden states $\mathbf{x}_1, \ldots, \mathbf{x}_L$ and $\hat{\mathbf{x}} = \mathbf{x}_L$ is the predicted solution.

As shown in Equation (11), the regularization employs a kinetic energy term and a potential energy term.

We now show the effect of the meta-data $\mathbf{f}_M$ in this regularizer. To this end, consider the minimization problem in Equation (6a), when there is no observed data $\mathbf{d}^{\text{obs}}$. That is, we only minimize the regularization term. Differentiating Equation (11) with respect to $\mathbf{x}$ yields the leapfrog scheme:

$$\mathbf{x}_{k+1} = 2\mathbf{x}_k - \mathbf{x}_{k-1} + \nabla_{\mathbf{x}_k} \mathbf{\Phi}(\mathbf{x}_k, \mathbf{f}_M, \mathcal{G}, \boldsymbol{\theta}_k). \tag{12}$$

Equation (12) is a residual GNN with two skip connections, also referred to as a neural network with hyperbolic dynamics (Ruthotto and Haber 2019), also studied for GNNs in Eliasof, Haber, and Treister (2021); Rusch et al. (2022). Thus, even without data $\mathbf{d}^{\text{obs}}$, if the meta-data $\mathbf{f}_M$ has sufficient information about the final state $\mathbf{x}$, this regularization can be sufficient to recover it. Having additional observed data on the final state $\mathbf{x}$ can further improve the recovery. In our experiments, we refer to this network as *Var-GNN*.

Inverse Scale-Space. The second technique we experiment with is a modification of a learned inverse scale-space technique proposed in (Eliasof, Haber, and Treister 2024) for solving inverse problems on a structured grid. We adapt it to operate on graph problems. The dynamical system in Equation (7) is simply discretized in two steps, obtaining:

$$\mathbf{z}_{k+\frac{1}{2}} = \mathbf{z}_k + \mu\mathbf{E}_k^\top \mathbf{J}_k^\top \left(\mathbf{d}^{\text{obs}} - F(\mathbf{E}_k\mathbf{z}_k, \mathcal{G})\right) \tag{13a}$$

$$\mathbf{z}_{k+1} = \mathbf{z}_{k+\frac{1}{2}} - s(\mathbf{z}_{k+\frac{1}{2}}, \mathbf{f}_M, \mathcal{G}, \boldsymbol{\theta}_k, t_k). \tag{13b}$$

For the embedding, $\mathbf{E}_k$, we use a linear layer and a standard multilayer GNN for s with time embedding. We refer to this method as *ISS-GNN*. Note that, similarly to Var-GNN, in the absence of data $\mathbf{d}^{\text{obs}}$, the effective network reduces to Equation (13b), which is a residual GNN. This implies that ISS-GNN can utilize the meta-data $\mathbf{f}_{\text{M}}$ to potentially recover the state $\mathbf{x}$, similar to standard uses of GNNs.

Proximal Methods. Another approach for solving inverse problems that blends scale-space and variational methods was proposed in (Adler and Öktem 2017; Mardani et al. 2018). Here, we modify the technique for graph learning.

In Proximal methods, the concept is to use proximal gradient descent as a numerical method for the solution of the optimization problem in Equation (6a). Such iteration reads:

$$\mathbf{x}_{k+\frac{1}{2}} = \mathbf{x}_k - \mu\mathbf{J}_k^\top (F(\mathbf{x}_k) - \mathbf{d}^{\text{obs}}), \tag{14a}$$

$$\mathbf{x}_{k+1} = \text{argmin}_{\mathbf{x}} \tfrac{1}{2}\|\mathbf{x} - \mathbf{x}_{k+\frac{1}{2}}\|^2 + \mathbf{R}(\mathbf{x}, \mathbf{f}_M, \boldsymbol{\theta}). \tag{14b}$$

This iteration can be computationally expensive due to the cost of solving the optimization problem in Equation (14). Therefore, Adler and Öktem (2017) and Mardani et al. (2018) propose to learn the solution of the optimization problem in Equation (14) with a network s that is:

$$s(\mathbf{x}_{k+\frac{1}{2}}, \mathbf{f}_M, \mathcal{G}, \boldsymbol{\theta}) = \text{argmin}_{\mathbf{x}}\big(\frac{1}{2}\|\mathbf{x} - \mathbf{x}_{k+\frac{1}{2}}\|^2 + \mathbf{R}(\mathbf{x}, \mathcal{G}, \mathbf{f}_{\mathcal{M}}, \boldsymbol{\theta})\big). \tag{15}$$

This approach leads to a two-stage network, where in the first step, a gradient step is applied (Equation (14a)), which aims to fit the data, and in the second step, a proximal step

is applied (Equation (15)). In Adler and Öktem (2017), the iteration in Equation (15) shares the same parameters $\boldsymbol{\theta}$ for every proximal step. In Mardani et al. (2018), the iteration is unrolled, to learn a parameter $\boldsymbol{\theta}_k$ per-iteration. This choice unties the link to variational methods but increases the flexibility of the method. In our experiments we follow Mardani et al. (2018), and modify the method proposed there to operate on graphs. We call this network *Prox-GNN*, which unites the two steps in Equation (14) as follows:

$$\mathbf{x}_{k+1} = s\left(\mathbf{x}_k - \mu \mathbf{J}_k^\top (F(\mathbf{x}_k) - \mathbf{d}^{\text{obs}}), \mathbf{f}_M, \mathcal{G}, \boldsymbol{\theta}_k\right), \quad (16)$$

where $s(\cdot)$ is a GNN that uses observed data $\mathbf{d}^{\text{obs}}$ and the meta-data to solve the problem. It is interesting to observe that, in the absence of observed data, we obtain a standard feed-forward GNN that uses the meta-data to predict $\mathbf{x}$.

5 Experiments

We evaluate the five models discussed in Section 4 on the diverse set of GRIPs discussed in Section 2. We utilize several datasets to demonstrate GRIPs, ranging from synthetic graphs (CLUSTER) to geometric datasets (ShapeNet), natural images (CIFAR10), road traffic (METR-LA), and spread of disease (Chickenpox-Hungary, shortened to CPOX). We provide full details on the datasets and our motivation for using them to demonstrate different problems, in Appendix D. We also provide a comprehensive description of the training, evaluation, and hyperparameter selection procedures for each problem in Appendix E, hyperparameter sensitivity results in Appendix F, and a discussion of the complexity and runtimes of the methods, in Appendix G. Our code is implemented in Pytorch (Paszke et al. 2017), and is available at https://github.com/nafi007/GraphInverseProblems.

Research Questions. In our experiments, we seek to address the following questions: (i) Can the framework proposed in this paper be applied to various GRIPs? (ii) Do neural methods consistently perform better than classical methods, and to what extent? (iii) What is the influence of using meta-data on downstream performance?

5.1 Property Completion

In the Property Completion problem, defined in Equation (2) and illustrated in Figure 1, for each graph, we randomly sample nb nodes per class, and the selected nodes' labels are the observed data. For example, in Figure 5, we show an example where the property to be recovered are the node labels, and only nb per class of them are available as input, besides possible meta-data such as coordinates. We present results with $nb = 4$ in Table 1, and results on $nb = 8, 16$ in Appendix F. We observe an accuracy-increasing trend is evident when increasing nb and a consistently better performance offered by neural approaches. Also, we study the effect of meta-data, as shown in Table 2 (and in Appendix F), where a significant drop in performance is seen in the models when meta-data was not used.

5.2 Inverse Source Estimation

We now discuss the results for the Inverse Source Estimation problem, which is described by the forward operator in Equation (3) and Figure 2. In this problem, node states $\mathbf{x}$ are diffused by k applications of the adjacency matrix where $k = 4, 8, 16$, and their diffused result is the observed data. As k grows, the problem becomes harder. The results with $k = 16$ are reported in on the CLUSTER dataset in Table 3, and on the CPOX dataset in Table 4, with additional results in Appendix F. We note that in the case of inverse source estimation, our ISS-GNN becomes similar to Huang et al. (2023). Our results indicate that all neural methods achieve better performance than classical methods.

Dataset	Model	Accuracy (%)↑	Data Fit (CE↓)
ShapeNet	LaplacianReg	74.90±0.00	3.87±0.00
	TikhonovReg	6.91±0.00	3.71±0.00
	Var-GNN	89.16±0.31	0.24±0.08
	ISS-GNN	89.59±0.53	0.27±0.02
	Prox-GNN	89.33±0.23	$2.94\pm2.8\cdot10^{-9}$
CLUSTER	LaplacianReg	70.13±0.00	1.78±0.00
	TikhonovReg	34.64±0.00	1.12±0.00
	Var-GNN	88.89±0.51	$1.04\pm4.9\cdot10^{-9}$
	ISS-GNN	57.08±7.36	$1.04\pm1.6\cdot10^{-8}$
	Prox-GNN	53.58±2.58	1.04±0.00

Table 1: Property Completion performance (accuracy±standard deviation (%)) on the ShapeNet and CLUSTER Datasets, with node budget per label per graph $nb = 4$. The Cross-Entropy (CE) loss between the observed and reconstructed data is shown for reference.

Meta-data	Model	Accuracy (%)↑	Data Fit (CE↓)
Yes	Var-GNN	89.16±0.31	0.24±0.08
No		26.89±0.02	0.37±0.02
Yes	ISS-GNN	89.59±0.53	0.27±0.02
No		26.45±0.02	$3.01\pm1.0\cdot10^{-6}$
Yes	Prox-GNN	89.33±0.23	$2.94\pm2.8\cdot10^{-9}$
No		26.96±0.01	$2.94\pm4.1\cdot10^{-9}$

Table 2: The impact of meta-data within different networks, on the Property Completion task on the ShapeNet datasets with node budget per label per graph $nb = 4$. The CE is between the observed and reconstructed data.

5.3 Inverse Graph Transport

This Inverse Graph Transport is described by Equation (4) and by Figure 3. We study the performance of the different models for path lengths $pl = 8, 16, 32$, meaning that the observed data at each node is an average of pl features of nodes that lie on a path that starts from the respective node (as further explained in Appendix D). The results with $pl = 32$ are shown in Table 5, indicating that neural models significantly outperform classical approaches, showing their problem efficacy at solving ill-posed problems.

5.4 Edge Property Recovery

We study a *nonlinear* inverse problem, differently than the previously presented linear problems. Here, the goal is to re-

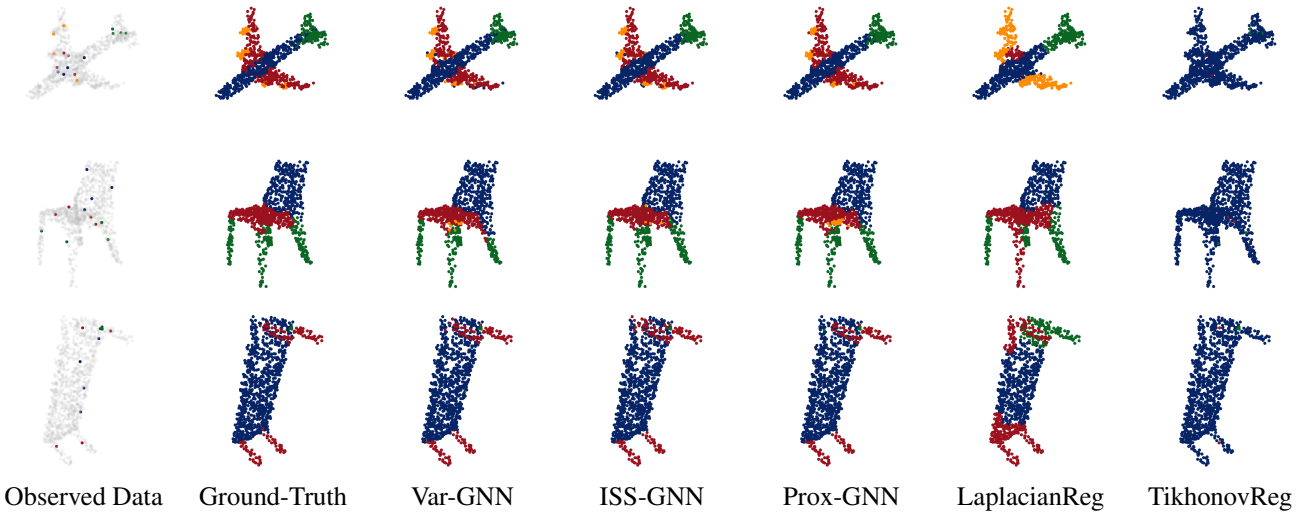

Figure 5: A qualitative comparison of different methods on the property completion problem on ShapeNet. Grey points in Observed Data indicate masked/unseen points. Neural approaches consistently outperform classical regularizers.

Method	Accuracy (%)↑	Data Fit (CE↓)
LaplacianReg	26.28±0.00	1.75±0.00
TikhonovReg	26.28±0.00	1.76±0.00
Var-GNN	82.36±0.20	$1.75 \pm 1.8 \cdot 10^{-9}$
ISS-GNN	52.31±4.18	$1.75 \pm 3.1 \cdot 10^{-9}$
Prox-GNN	44.48±3.63	$1.75 \pm 3.8 \cdot 10^{-9}$

Table 3: Inverse Source Estimation accuracy±standard deviation (%) on CLUSTER dataset, with $k = 16$ diffusion steps. CE is between the observed and reconstructed data.

Method	nMSE↓	Data Fit (nMSE↓)
LaplacianReg	0.84±0.00	0.048±0.00
TikhonovReg	0.84±0.00	0.051±0.00
Var-GNN	0.59±0.01	$5.9 \cdot 10^{-11} \pm 0.00$
ISS-GNN	$0.59 \pm 8.4 \cdot 10^{-5}$	$5.5 \cdot 10^{-11} \pm 0.00$
Prox-GNN	0.73±0.00	$1.0 \cdot 10^{-5} \pm 0.00$

Table 4: Inverse Source Estimation normalized-mean-squared error (nMSE) of the error and data-fit on the Chickenpox-Hungary dataset, with $k = 16$ diffusion steps.

cover edge weights given data that was diffused using these weights, as described in Equation (5) and by Figure 4. Results on the superpixels CIFAR10 dataset are shown in Table 6, where we see that classical methods struggle to offer a low relative error, compared with 5 times lower (better) error given by Var-GNN and ISS-GNN. The results on different GRIPs and datasets show the generality of the framework.

6 Conclusions and Discussion

In this paper, we propose a framework for Graph Inverse Problems. We show that while GRIPs can have very different forward problems, the recovery process can be addressed in a very similar fashion. We have experimented with a number of architectures for the solution of the problem, ranging from classical ones to learned ones. We observe that learning the regularization typically improves over unlearned regularization techniques. One important feature of GRIPs is the varying graph structure, and the second is the presence of meta-data. While meta-data is common for GRIPs, it is much less common for other inverse problems. We also show that using learned graph-based regularizations provides a natural way to include meta-data in the inversion process, leading to improved solutions.

Model	nMSE↓	Data Fit (nMSE↓)
LaplacianReg	0.61±0.00	0.11±0.00
TikhonovReg	0.36±0.00	0.04±0.00
Var-GNN	$0.004 \pm 2.0 \cdot 10^{-5}$	$8.2 \cdot 10^{-6} \pm 0.00$
ISS-GNN	$0.21 \pm 5.9 \cdot 10^{-5}$	$0.004 \pm 7.9 \cdot 10^{-6}$
Prox-GNN	$0.21 \pm 3.5 \cdot 10^{-3}$	$0.003 \pm 1.4 \cdot 10^{-4}$

Table 5: Inverse Graph Transport normalized-mean-squared-error (nMSE) on the METR-LA dataset with $pl = 32$.

Model	nMSE↓	Data Fit (nMSE↓)
LaplacianReg	0.598±0.00	0.0697±0.00
TikhonovReg	0.626±0.00	0.0581±0.00
Var-GNN	0.114±0.03	$0.073 \pm 2.4 \cdot 10^{-5}$
ISS-GNN	0.122±0.02	$0.078 \pm 3.9 \cdot 10^{-5}$
Prox-GNN	0.434±0.02	$0.023 \pm 6.8 \cdot 10^{-6}$

Table 6: Edge Property Recovery results on the CIFAR10 (superpixels) dataset.

Acknowledgments

ME is funded by the Blavatnik-Cambridge fellowship, the Accelerate Programme for Scientific Discovery, and the Maths4DL EPSRC Programme.

References

Adler, J.; and Öktem, O. 2017. Solving ill-posed inverse problems using iterative deep neural networks. *Inverse Problems*, 33(12): 124007.

Bronstein, M. M.; Bruna, J.; Cohen, T.; and Veličković, P. 2021. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*.

Bronstein, M. M.; Bruna, J.; LeCun, Y.; Szlam, A.; and Vandergheynst, P. 2017. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4): 18–42.

Burger, M.; Gilboa, G.; Osher, S.; and Xu, J. 2006. Nonlinear inverse scale space methods. *Commun. Math. Sci.*, 4(1): 179–212.

Calvetti, D.; Reichel, L.; and Shuibi, A. 2005. Invertible smoothing preconditioners for linear discrete ill-posed problems. *Applied numerical mathematics*, 54(2): 135–149.

Calvetti, D.; and Somersalo, E. 2005. Priorconditioners for linear systems. *Inverse Problems*.

Chen, Z.; Li, L.; and Bruna, J. 2019. Supervised Community Detection with Line Graph Neural Networks. In *International Conference on Learning Representations*.

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*, 3844–3852.

Eliasof, M.; and Haber, E. 2024. Graph Neural Networks for Binary Programming. *arXiv preprint arXiv:2404.04874*.

Eliasof, M.; Haber, E.; and Treister, E. 2021. PDE-GCN: Novel architectures for graph neural networks motivated by partial differential equations. *Advances in Neural Information Processing Systems*, 34: 3836–3849.

Eliasof, M.; Haber, E.; and Treister, E. 2023. DRIP: Deep Regularizers for Inverse Problems. *arXiv preprint arXiv:2304.00015*.

Eliasof, M.; Haber, E.; and Treister, E. 2024. An Over Complete Deep Learning Method for Inverse Problems. *arXiv preprint arXiv:2402.04653*.

Engl, H. W.; Hanke, M.; and Neubauer, A. 1996. *Regularization of inverse problems*, volume 375. Springer Science & Business Media.

Golub, G.; and Loan, C. V. 1983. *Matrix Computations*. Johns Hopkins University Press.

Hansen, P. C. 1997. *Rank-Deficient and Discrete Ill-Posed Problems*. Philadelphia: SIAM.

Huang, B.; Yu, W.; Xie, R.; Xiao, J.; and Huang, J. 2023. Two-stage Denoising Diffusion Model for Source Localization in Graph Inverse Problems. arXiv:2304.08841.

Jiang, D.; Wu, Z.; Hsieh, C.-Y.; Chen, G.; Liao, B.; Wang, Z.; Shen, C.; Cao, D.; Wu, J.; and Hou, T. 2021. Could graph neural networks learn better molecular representation for drug discovery? A comparison study of descriptor-based and graph-based models. *Journal of cheminformatics*, 13: 1–23.

Khemani, B.; Patil, S.; Kotecha, K.; and Tanwar, S. 2024. A review of graph neural networks: concepts, architectures, techniques, challenges, datasets, applications, and future directions. *Journal of Big Data*, 11(1): 18.

Kipf, T. N.; and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Ling, C.; Chowdhury, T.; Ji, J.; Li, S.; Züfle, A.; and Zhao, L. 2024. Source Localization for Cross Network Information Diffusion. *arXiv preprint arXiv:2404.14668*.

Manríquez, R.; Guerrero-Nancuante, C.; Martínez, F.; and Taramasco, C. 2021. Spread of Epidemic Disease on Edge-Weighted Graphs from a Database: A Case Study of COVID-19. *International Journal of Environmental Research and Public Health*, 18(9).

Mardani, M.; Sun, Q.; Donoho, D.; Papyan, V.; Monajemi, H.; Vasanawala, S.; and Pauly, J. 2018. Neural proximal gradient descent for compressive imaging. *Advances in Neural Information Processing Systems*, 31.

Nadler, B.; Srebro, N.; and Zhou, X. 2009. Statistical Analysis of Semi-Supervised Learning: The Limit of Infinite Unlabelled Data. In Bengio, Y.; Schuurmans, D.; Lafferty, J.; Williams, C.; and Culotta, A., eds., *Advances in Neural Information Processing Systems*, volume 22. Curran Associates, Inc.

Nagy, J.; and Hansen, P. 2006. *Deblurring Images*. Philadelphia: SIAM.

Natterer, F. 2001. *The mathematics of computerized tomography*. Society for Industrial Mathematics.

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic differentiation in PyTorch. In *Advances in Neural Information Processing Systems*.

Rudin, L.; Osher, S.; and Fatemi, E. 1992. Nonlinear total variation based noise removal algorithms. In *Proceedings of the eleventh annual international conference of the Center for Nonlinear Studies on Experimental mathematics : computational issues in nonlinear science*, 259–268. Elsevier North-Holland, Inc.

Rusch, T. K.; Chamberlain, B.; Rowbottom, J.; Mishra, S.; and Bronstein, M. 2022. Graph-coupled oscillator networks. In *International Conference on Machine Learning*, 18888–18909. PMLR.

Ruthotto, L.; and Haber, E. 2019. Deep neural networks motivated by partial differential equations. *Journal of Mathematical Imaging and Vision*, 1–13.

Scarselli, F.; Gori, M.; Tsoi, A. C.; Hagenbuchner, M.; and Monfardini, G. 2008. The graph neural network model. *IEEE transactions on neural networks*, 20(1): 61–80.

Schuetz, M. J.; Brubaker, J. K.; and Katzgraber, H. G. 2022. Combinatorial optimization with physics-inspired graph neural networks. *Nature Machine Intelligence*, 4(4): 367–377.

Tarantola, A. 1987. *Inverse problem theory*. Elsevier, Amsterdam.

Tenorio, L.; Andersson, F.; de Hoop, M.; and Ma, P. 2011. Data analysis tools for uncertainty quantification of inverse problems. *Inverse Problems*, 045001.

Tikhonov, A. N. 1950. Determination of the electrical characteristics of the deep strata of the earth's crust. *Doklady Akadamia Nauk*, 73: 295–297.

Tikhonov, A. N.; and Arsenin, V. Y. 1977. *Solutions of Ill-posed Problems*. Winston.

Yan, X.; Fang, H.; and He, Q. 2023. Diffusion Model for Graph Inverse Problems: Towards Effective Source Localization on Complex Networks. In Oh, A.; Naumann, T.; Globerson, A.; Saenko, K.; Hardt, M.; and Levine, S., eds., *Advances in Neural Information Processing Systems*, volume 36, 22326–22350. Curran Associates, Inc.

Zhou, D.; Bousquet, O.; Lal, T. N.; Weston, J.; and Schölkopf, B. 2004. Learning with local and global consistency. *Advances in neural information processing systems*, 16: 321–328.