# Structural Landmarking and Interaction Modelling: on Resolution Dilemmas in Graph Classification

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

Graph neural networks are promising architecture for learning and inference with graph-structured data. Yet difficulties in modelling the "parts" and their "interactions" still persist in terms of graph classification, where graph-level representations are usually obtained by squeezing the whole graph into a single vector through graph pooling. From complex systems point of view, mixing all the parts of a system together can affect both model interpretability and predictive performance, because properties of a complex system arise largely from the interaction among its components. We analyze the intrinsic difficulty in graph classification under the unified concept of "resolution dilemmas" with learning theoretic recovery guarantees, and propose "*SLIM*", an inductive neural network model for Structural Landmarking and Interaction Modelling. It turns out, that by solving the resolution dilemmas, and leveraging explicit interacting relation between component parts of a graph to explain its complexity, *SLIM* is more interpretable, accurate, and offers new insight in graph representation learning.

## 1 Introduction

Complex systems are ubiquitous phenomenon in natural and scientific disciplines, and how relationships between parts give rise to global behaviours of a system is a central theme in many areas of study such as system biology [5], neural science [16], and drug and material discoveries [31] [27].

Graph neural networks are promising architecture for representation learning on graphs - the structural abstraction of complex system. State-of-the-art performance is observed in various graph mining tasks [4, 10, 14, 37, 32, 23, 35, 43]. However, due to the non-Euclidian nature, challenges still exist in graph classification. For example, in order to generate a fixed-dimensional graph-level representation, GNN combines information from each node through *graph pooling*. In combined forms, a graph will collapse into a "super-node", where identities of the constituent sub-graphs and their inter-connections are mixed together. Is this the best way to generate graph-level features? From complex systems view, mixing all parts of a system can affect interpretability and model prediction, because properties of a complex system arise largely from the *interactions* among its components [15, 9, 7].

The choice of the "collapsing"-style graph pooling roots deeply in the lack of natural alignment among graphs that are not isomorphic. Therefore the pooling sacrifices structural details for feature compatibility. In recent years, substructure patterns draw considerable attention in graph mining, such as motifs [22, 1, 34, 2] and graphlets [29]. It provides an intermediate scale for structure comparison or counting, and has been considered in node embedding [18], deep graph kernels [38] and graph convolution [39]. However, due to the combinitorial nature, only substructures of very small sizes (4 or 5 nodes) can be considered [38, 34], greatly limiting the coverage of structural variations; also, handling substructures as discrete objects makes it difficult to compensate for their similarities, at least computationally, and so the risk of overfit may rise in supervised learning scenarios.

Submitted to 34th Conference on Neural Information Processing Systems (NeurIPS 2020). Do not distribute.

These intrinsic difficulties are related to the concept of *resolution* in graph-structured data processing. Resolution is the scale at which measurements can be made and/or information processing algorithms are conducted. Here, we will first define two relevant terms, i.e., the spatial resolution and the structural resolution, and how they may affect the performance of graph classification.

First, *spatial resolution* is related to the geometrical scale of the "elementary component" of a graph on which an algorithm operates. It can range from nodes, to sub-graphs, or entire graph. Graph details beyond effective spatial resolution are algorithmically unidentifiable. For example, graph pooling compresses the whole graph into a single vector, and so the spatial resolution drops to the lowest: node and edge identities are mixed together, and subsequent classification layer can no longer exploit any substructure or their connections, but just a global aggregation. We call this **vanishing spatial resolution**. Insufficient spatial resolution may affect the interpretability, and also the predictive power since global property of a complex system arises largely from the its inherent interactions [15, 9, 7].

Second, *structural resolution* is the fineness level in differentiating between substructures. substructures (or sub-graphs) shed light on the functional organization and graph alignment. However, they are treated in a discrete, and over-delicate manner: two substructures may be considered distinct even if they share significant similarity. We call it **exploding structural resolution**. It can lead to the risk of overfitting, similar to observed in deep graph kernels [38] and dictionary learning [19].

We believe that both resolution dilemmas originate from the way we perform profiling, identification, and alignment of substructures. Substructures are building blocks of a graph; relations like interaction or alignment are all defined between substructures (of varying scales). However, exact substructure matching is too costly and prone to overfit, leading to exploding structural resolution; meanwhile, graph alignment becomes infeasible when substructure matching is poorly defined, and so collapsing-style graph pooling becomes the norm, which finally leads to vanishing spatial resolution.

**Our contribution**. In this paper, we propose a simple neural architecture called "Structural Landmarking and Interaction Modelling" - or SLIM, for inductive graph classification. The key idea is to embed substructure instances into a continuous metric space and learn structural landmarks there for explicit interaction modelling. The SLIM network can effectively resolve the resolution dilemmas. More importantly, by fully exploring the diverse structural distribution of the input graphs, any substructure instance and even unseen examples can be mapped parametrically to a common and optimizable structural landmark set. This enables a novel, *identity-preserving graph pooling* paradigm, where the interacting relation between constituent parts of a graph can be modelled explicitly, shedding important light on the functional organizations of complex systems.

The design philosophy of SLIM comes from the long-standing views of complex systems: complexity arises from interaction. Therefore, explicit modelling of the parts and their interactions is key to explaining the complexity and improving the prediction. In contrast, graph neural networks is more about "integration", where delicate part-modelling like convolution does exist but finally obscured in the pooling process. It turns out, that by respecting the structural organization of complex systems, SLIM is more interpretable, accurate, and provides new insights in graph representation learning.

We will discuss the resolution dilemmas and related works in Section 2. Section 3, 4 and 5 covers the design, analysis, and performance of SLIM, respectively. The last section concludes the paper.

## 2 Resolution Dilemmas in Graph Classification

A complex system is composed of many parts that interact with each other in a non-simple way. Since graphs are structural abstraction of complex systems, accurate graph classification depends on how global properties of a system relate to its structure. It is believed that the property (and complexity) of a complex system arises from the interaction among its components [9, 7]. So, accurate interaction modelling should benefit prediction. However, this is non-trivial due to resolution dilemmas.

### 2.1 Spatial Resolution Diminishes in Graph Pooling

Graph neural networks (GNN) for graph classification typically has two stages: graph convolution and graph pooling [14, 37]. The spatial resolutions for these two stages are significantly different.

The goal of convolution is to pass message among neighboring nodes in the general form of $h_v = \texttt{AGGREGATE}\left(\{h_u, u \in \mathcal{N}_v\}\right)$, where $\mathcal{N}_v$ is the neighbors of $v$ [14, 37]. Here, the spatial resolution is

controlled by the number of convolution layers: more layers capture lager substructures/sub-trees and can lead to improved discriminative power [37]. In other words, a medium resolution (substructure level) can be more informative functional markers than a high resolution (node level). In practice, multiple resolutions can be combined via `CONCATENATE` function [14, 37] for subsequent processing.

The goal of graph pooling is to generate compact, graph-level representations that are compatible across graphs. Due to the lack of natural alignment between graphs that are not isomorphic, graph pooling typically "squeezes" a graph $\mathcal{G}$ into a single vector (or "super-node") in the form of $h_{\mathcal{G}} = \texttt{READOUT}\left(\{f(h_v), \forall v \in \mathcal{V}\}\right)$, where $\mathcal{V}$ is the node of $\mathcal{G}$. Different readout functions have been proposed, including max-pooling [6], sum-pooling [37], various pooling functions (`MEAN`, `LSTM`, etc.) [14], or deep sets [41]; attention has been used to evaluate node importance in attention pooling [17] and gPool [13]; besides, hierarchical differential pooling has also been investigated [40].

An important resolution bottleneck occurs in graph pooling, as shown in Figure 1. Since all the nodes are mixed into one, subsequent classifier can no longer identify any individual substructure nor their interactions, regardless of the resolution in graph convolution. We call this "diminishing spatial resolution", which can be undesirable[1] in that: (1) how much information in well-designed convolution domain can penetrate through the pooling layer for final prediction is hard to analyze/control; (2) in molecule classification, graph labels hinge on functional modules and how they organize [31]; an overly coarse spatial resolution will mix up functional modules and conceal their interaction.

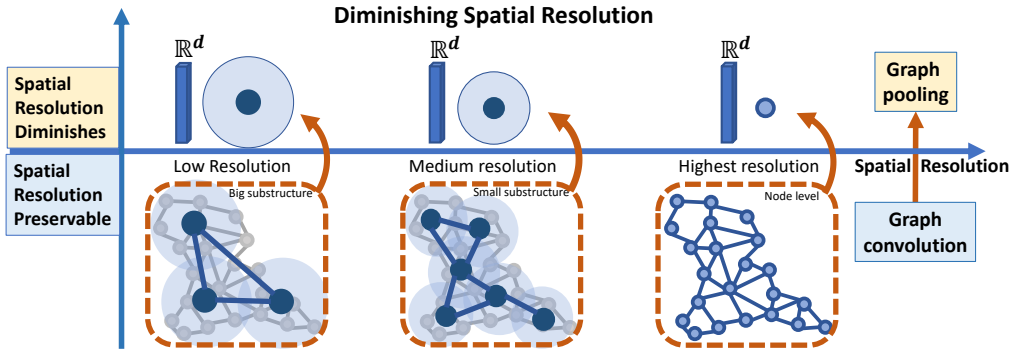



Figure 1: Spatial resolution vanishes after graph pooling. (Note: not all nodes are marked with convolution - the shaded circles; see Appendix Sec 8.4 for more discussion on relation with hierarchical processing.)

Can meaningful spatial resolution(s) survive graph pooling? The answer is yes. Indeed, it involves substructure alignment, and the notion of structural resolution. See discussions below.

### 2.2 Structural Resolution Explodes in Substructure Identification

Substructures are the basic unit to accommodate interacting relations. A global criteria to identify and align substructures is the key to preserving substructure identities and comparing the inherent interactions across graphs. Again, the fineness level in determining whether two substructures are "similar" or "different" is subject to a wide spectrum of choices, which we call "structural resolution".

We illustrate in Figure 2. The right end denotes the finest resolution in differentiating between substructures: exact matching, as we manipulate motif/graphlet [22, 1, 34, 39, 29]. The exponential configuration of sub-graphs will finally lead to an "exploding" structural resolution, because maintaining a large number of unique substructures is infeasible and easily overfits. The left end of the spectrum treats all substructures the same and underfits the data. We are interested in a medium structural resolution, where similar substructures are mapped to the same identity, which we believe can benefit the generalization performance (see Figure 4 for empirical evidence).

Theoretically, an over-delicate structural resolution corresponds to a highly "coherent" basis in representing a graph, leading to unidentifiable dictionary learning [11, 20]. Structural landmarking is exactly aimed at controlling structural resolution and improve incoherence for graph classification.

[1] Some work adopt different aggregation strategies: Sortpooling arranges nodes in a linear chain and perform 1d-convolution [42]; SEED uses distribution of multiple random walks [33]; Deep graph kernel evaluates graph similarity by subgraph counts [38]. Explicit modelling of the interaction between graph parts is not considered.

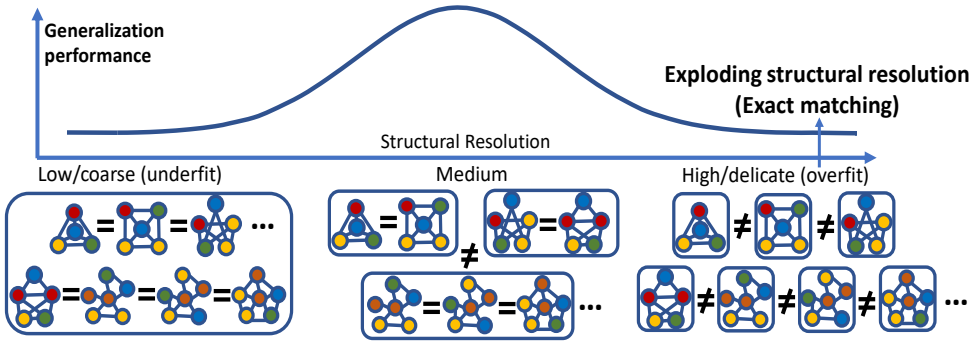



Figure 2: How structural resolution may affect the generalization performance. Only small substructures here for illustration; node types do make a difference in profiling the substructures.

## 3 Structural Landmarking and Interaction Modelling (SLIM)

Considering the difficulty in manipulating substructures as discrete objects, we embed them in a continuous space, and transform all structure-related operations from discrete and off-the-shelf version to continuous and optimizable counterpart. The key idea of SLIM is the identification of structural landmarks in this new space, via both unsupervised compression and supervised fine-tuning, through the distribution of embedded substructures under possibly multiple scales. Structural landmarking resolves resolution dilemmas and allow explicit interaction modelling in graph classification.

**Problem Setting**. Give a set of labeled graphs $\{\mathcal{G}_i, y_i\}$'s for $i = 1, 2, ..., n$, with each graph defined on the node/edge set $\mathcal{G}_i = (\mathbf{V}_i, \mathbf{E}_i)$ with adjacency matrix $\mathbf{A}_i \in \mathbb{R}^{n_i \times n_i}$ where $n_i = |\mathbf{V}_i|$, and $y_i \in \{\pm 1\}$. Assume that nodes are drawn from $c$ categories, and the node attribute matrix for $\mathcal{G}_i$ is $\mathbf{X}_i \in \mathbb{R}^{n_i \times c}$. Our goal is to train an inductive model to predict the labels of the testing graphs.

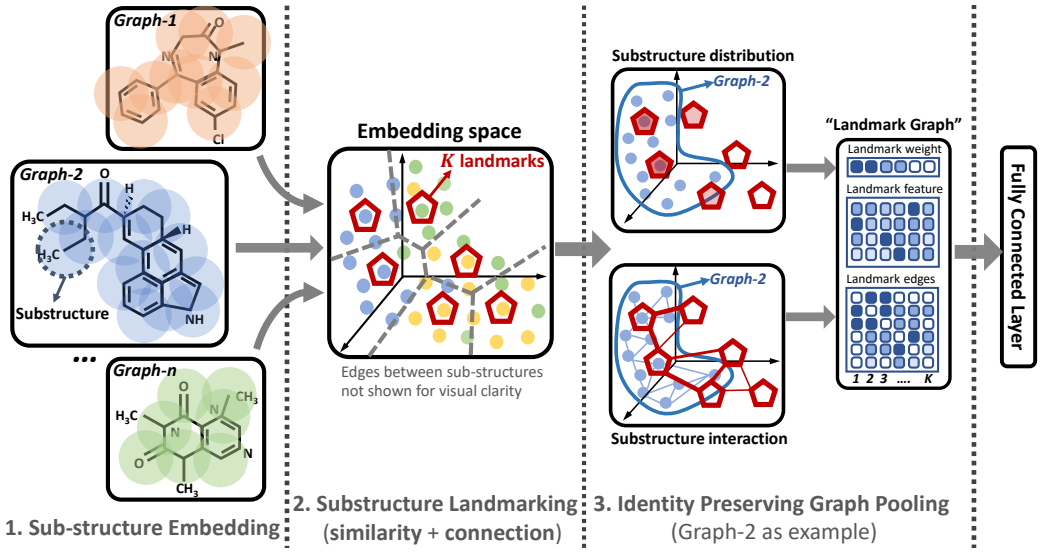



Figure 3: The three main steps of the SLIM network illustrated in molecule graph classification.

The SLIM network has three main steps: (1) sub-sturcture embedding, (2) substructure landmarking, and (3) identity-preserving graph pooling, as shown in Figure 3. Detailed discussion follows.

### 3.1 Substructure Embedding

The goal of substructure embedding is to extract substructure instances and embed them in a metric space. One can employ multiple layers of convolutions [14, 37] to model substructures (rooted sub-trees), or randomly sample sub-graphs [29]. For convenience, we simply extract one sub-graph

instance from each node using a $k$-hop breath-first search, which controls the spatial resolution[2]. In Figure 3, sub-graphs in the shaded circles around each atom is a substructure instance.

Let $\mathbf{A}_i^{(k)}$ be the $k$th-order adjacency matrix, i.e., the $pq$th entry equals 1 only if node $p$ and $q$ are within $k$-hops away. Since each sub-graph is associated with one node, the sub-graphs extracted from $\mathcal{G}_i$ can be represented as $\mathbf{Z}_i = \mathbf{A}_i^{(k)}\mathbf{X}_i$, whose $j$th row is a $c$-dimensional vector summarizing the counts of the $c$ node-types in the sub-graph around the $j$th node. Variations include (1) emphasize the center node, $\mathbf{Z}_i = [\mathbf{X}_i;\ \mathbf{A}_i\mathbf{X}_i]$; (2) layer-wise node distribution $\mathbf{Z}_i = [\tilde{\mathbf{A}}_i^{(1)}\mathbf{X}_i;\ \tilde{\mathbf{A}}_i^{(2)}\mathbf{X}_i;\ ...\ \tilde{\mathbf{A}}_i^{(k)}\mathbf{X}_i]$, where $\tilde{\mathbf{A}}_i^{(k)}$ specifies whether two nodes in $\mathcal{G}_i$ are *exactly* $k$-hops away; or (3) weighted Layer-wise summation $\mathbf{Z}_i = \alpha_k \sum_k \tilde{\mathbf{A}}_i^{(k)}\mathbf{X}_i$, where $\alpha_k$'s are non-negative weighting that decays with $k$.

Next we consider embedding the substructure instances (i.e., rows of $\mathbf{Z}_i$'s) into a latent space so that statistical manipulations can better align with the prediction task. The embedding should preserve important proximity relations to facilitate subsequent landmarking: if two substructures are similar, or they often inter-connect with each other, their embedding should be close. In other words, the embedding should be smooth regard to both structural similarities and geometrical interactions.

A parametric transform on $\mathbf{Z}_i$'s with controlled complexity can guarantee the smoothness of embedding w.r.t. structural similarity, e.g., an autoencoder $f(\mathbf{Z}_i) = \sigma\left(\sigma\left(\mathbf{Z}_i\mathbf{T}_1 + \mathbf{b}_1\right)\mathbf{T}_2 + \mathbf{b}_2\right)$. Let $\mathbf{H}_l = f(\mathbf{Z}_i) \in \mathbb{R}^{n_l \times d}$ be the embedding of the $n_l$ sub-graph instances extracted from $\mathcal{G}_l$. To maintain the smoothness of $\mathbf{H}_i$'s w.r.t. geometric interaction, we will maximize the log-likelihood of the co-ouuurrence of substructure instances in each graph, similar to word2vec [21]

$$\max \sum_{l=1}^{n}\sum_{i=1}^{n_l}\sum_{j\in\mathcal{N}_i^l} \log\left(\frac{\exp\langle \mathbf{H}_l(i,:), \mathbf{H}_l(j,:)\rangle}{\sum_{j'}\exp\langle \mathbf{H}_l(i,:), \mathbf{H}_l(j',:)\rangle}\right) \tag{1}$$

Here $\mathbf{H}_l(i,:)$ is the $l$th row of $\mathbf{H}_l$, $\langle,\rangle$ is inner product, and $\mathcal{N}_i^l$ are the neighbors of node-$i$ in graph $\mathcal{G}_l$. This loss function tends to embed strongly inter-connecting substructures close to each other.

### 3.2 Substructure Landmarking

The goal of structural landmarking is to identify a set of informative structural landmarks in the continuous embedding space which has: (1) high statistical coverage, namely, the landmarks should faithfully recover distribution of the substructures from the input graphs, so that we can generalize to new substructure examples from the distribution; and (2) high discriminative power, namely the landmarks should be able to reflect discriminative interaction patterns for classification.

Let $\mathbf{U} = \{\boldsymbol{\mu}_i, \boldsymbol{\mu}_2, ..., \boldsymbol{\mu}_K\}$ be the structural landmarks. In order for them to be representative of the substructure distribution, it is desirable that each sub-graph instance is faithfully approximated with the closest landmark. We will minimize the following distortion loss

$$\sum_{i=1}^{n}\sum_{j=1}^{n_i} \min_{k=1,2,...,K} \|\mathbf{H}_i(j,:) - \boldsymbol{\mu}_k\|^2. \tag{2}$$

Here $\mathbf{H}_i(j,:)$ denotes the $j$th row (substructure) from graph $\mathcal{G}_i$. In practice, we will implement a soft assignment by using one cluster indicator matrix $\mathbf{W}_i \in \mathbb{R}^{n_i \times k}$ for each graph $\mathcal{G}_i$, whose $jk$-th entry is the probability that the $j$th substructure of $\mathcal{G}_i$ belongs to the $k$th landmark $\boldsymbol{\mu}_k$. Inspired by deep embedding clustering [36], $\mathbf{W}_i$ is parameterized by a Student's t-distribution

$$\mathbf{W}_i(j,k) = \frac{\|(1 + \mathbf{H}_i(j,:) - \boldsymbol{\mu}_k\|^2/\alpha)^{-\frac{\alpha+1}{2}}}{\sum_{k'}\|(1 + \mathbf{H}_i(j,:) - \boldsymbol{\mu}'_k\|^2/\alpha)^{-\frac{\alpha+1}{2}}},$$

and the loss function can be greatly simplified by minimizing the KL-divergence

$$\min_{\mathbf{U},\mathbf{H}'_\mathbf{i}\mathbf{s}} \sum_i \mathrm{KL}\left(\mathbf{W}_i, \tilde{\mathbf{W}}_i\right), \quad \text{s.t. } \tilde{\mathbf{W}}_i(j,k) = \frac{\mathbf{W}_i^2(j,k)/\sum_l \mathbf{W}_i(l,k)}{\sum_{k'}\left[\mathbf{W}_i^2(j,k')/\sum_l \mathbf{W}_i(l,k')\right]}. \tag{3}$$

Here, $\tilde{\mathbf{W}}_i$ is a self-sharprening version of $\mathbf{W}_i$, and minimizing the KL-distance forces each substructure instance to be assigned to only a small number of landmarks similar to sparse dictionary learning. Besides the unsupervised regularization in (2) or (3), learning of the structural landmarks will also be affected by the classification loss, guaranteeing the discriminative power of the landmarks.

[2]When $k$ is large, one subgraph around each node may be unnecessary. See discussion in Appendix (Sec8.4).

### 3.3 Identity-Preserving Graph Pooling

The goal of identity-preserving graph pooling is to project structural details of each graph onto the common space of landmarks, so that a compatible, graph-level feature can be obtained that simultaneously preserves the identity of the parts (substructures) and models their interactions.

The structural landmarking mechanism allows computing rich graph-level features. First, we can model substructure distributions. The density of the $K$ substructure landmarks in graph $\mathcal{G}_i$ can be computed as $\mathbf{p}_i = \mathbf{W}_i' \cdot \mathbf{1}_{n_i \times 1}$. Furthermore, the first-order moment of substructures belonging to each of the $K$ landmarks in $\mathcal{G}_i$ is $\mathbf{M}_i = \mathbf{X}_i' \cdot \mathbf{W}_i \cdot \mathbf{P}_i^{-1}$ where $\mathbf{P}_i = \text{diag}(\mathbf{p}_i)$, and the $k$th column of $\mathbf{M}_i$ is the mean of $\mathcal{G}_i$'s substructure instances belonging to the $k$th landmark. Second, we can model how the $K$ landmarks interact with each other in graph $\mathcal{G}_i$. To do this, we can project the adjacency matrices $\mathbf{A}_i$'s onto the landmark sets and obtain a $\mathbb{R}^{K \times K}$ interaction matrix $\mathbf{C}_i = \mathbf{W}_i \cdot \mathbf{A}_i \cdot \mathbf{W}_i'$, which encodes the interacting relations (geometric connections) among the $K$ structural landmarks.

These features can be combined together for final classification. For example, they can be reshaped and concatenated to feed into the fully-connected layer. One can also resort to more intuitive ways; for example, using first-order and second-order features together, one can transform each graph $\mathcal{G}_i$ into a constant-sized, "landmark" graph with node feature $\mathbf{M}_i$, node weight $\mathbf{p}_i$, and edge weights $\mathbf{C}_i$. Then standard graph convolution can be applied on the landmark graphs to generate graph-level features (without pains of graph alignment anymore). In experiments, for simplicity, we will compute the normalized interaction matrix $\tilde{\mathbf{C}}_i = \mathbf{P}_i^{-1}\mathbf{C}_i\mathbf{P}_i^{-1}$ and use it as features, which works pretty well on all the benchmark datasets. More detailed discussion can be found in Appendix (Sec 8.7).

## 4 Theoretic Analysis and Discussions

We provide learning theoretic support on the choice of structural resolution (landmark size $K$). Graphs are bags of inter-connected substructure instances, and each instance $\mathbf{z}$ can be represented by the landmarks as $\mathbf{z} = \sum_{k=1}^{K} \alpha_k \boldsymbol{\mu}_k$. A too small number of landmarks fails to recover basic data structures, whereas too many landmarks will result in overfitting (e.g. in exact substructure matching where a maximal $K$ is used for reconstruction) [19]. In dictionary learning, the mutual coherence is a crucial index in evaluating the redundancy of the code-vectors, which is defined as

$$\mu(\mathbf{U}) = \max_{i,j} \left| \langle \boldsymbol{\mu}_i, \boldsymbol{\mu}_j \rangle \right|, \tag{4}$$

where $\langle \cdot, \cdot \rangle$ denotes the normalized correlation. A lower self-coherence permits better support recovery [11]; while large coherence leads to worse stability in both sparse coding and classification [20]. In particular, a faithful recovery of the sparse signal support is guaranteed only when

$$|\alpha|_0 \leq \frac{1}{2}\left(1 + \frac{1}{\mu(\mathbf{U})}\right). \tag{5}$$

Obviously, large $\mu(\mathbf{U})$ leads to unstable solutions. In the following, we quantify a lower-bound of the coherence as a factor of the landmark size $K$ in clustering-based basis selection, since the sparse coding and $k$-means algorithm generate very similar code vectors [8].

**Theorem 1.** *The lower bound of the squared mutual coherence of the landmark vectors increases monotonically with $K$, the number of landmarks in clustering-based sparse dictionary learning.*

$$\mu^2(\mathbf{U}) \quad \geq \quad 1 - \frac{4C_dC_p}{u_{max}^2 K^{\frac{1}{d}}} \left( \left\lfloor \left(\frac{K}{2}\right)^{\frac{1}{d}} \right\rfloor^{-1} + 1 \right)$$

*Here, $d$ is the dimension, $C_d = \frac{3}{2}\left(1 + \log(d)/d\right)\gamma_d V_d$, where $\gamma_d = 1 + d\log(d\log(d))$ and $V_d = 2\Gamma(\frac{1}{2})^d / d\Gamma(\frac{d}{2})$ is the volume of the $d$-dimensional unit ball; $u_{max}$ is the maximum $\ell_2$-norm of (a subset) of the landmark vectors $\boldsymbol{\mu}_k$'s, and $C_p$ is a factor depending on data distribution $p(\cdot)$.*

Proof is in Appendix (Sec 8.1). Theorem 1 says that when the landmark set size $K$ increases, the mutual coherence has a lower bound that consistently increases and violates the recovery condition (5). In fact, a very high structural resolution (like exact matching) leaves a heavy burden to subsequent classifiers by failing to compensate for structural similarities. This justifies the SLIM network where the landmark set size can be controlled conveniently to avoid unstable dictionary learning.

**Discussions**. GNNs have shown great potential in graph isomorphism test by generating injective graph embedding, thanks to the theoretic foundations [37, 23]. However, accurate graph classification needs more thought: classification is not injective; besides, quality of features is also of notable importance. SLIM provides new insight in both respects: (1) it finds a tradeoff in the duality of handling similarity and distinctness; (2) it explores new ways of generating graph-level features: instead of aggregating all parts together as in GNNs, it taps into the vision of complex systems so that interaction between the parts is leveraged to explain the complexity and improve the learning. More discussions are in Appendix (Sec 8.2-8.8), including the choice of spatial/structural resolutions, interpretability, hierarchical and semi-supervised version, and comparison with graph kernels.

# 5 Experiments

**Benchmark data**. We have used a number of popular benchmark data sets for graph classification. (1) MUTAG: chemical compound data set with 188 instances and two classes; there are 7 node/atom types, and 3 edge/bound types (bond types are ignored). (2) PROTEINS: protein molecule data set with 1113 instances and three classes; there are 3 node types (secondary structure elements). NCI1: chemical compounds data set for cancer cell lines with 4110 instances and two classes. (4) PTC: chemical compound data set for toxicology prediction with 417 instances and 8 classes. (5) D&D data set for enzyme classification with 1178 instances and two classes.

**Competing methods**. We have incorporated a number of highly competitive methods proposed in recent years for comparison: (1) Graph neural tangent kernel (GNTK) [12]; (2) Graph Isomorphism Network (GIN) [37]; (3) End-to-end graph classification (DCGNN) [42]; (4) Hierarchical and differential pooling (DiffPool) [40]; (5) Self-attention Pooling (SAG) [17]; (6) Convolutional network for graphs (PATCHY-SAN) [25]; (7) Graphlet kernel (GK) [30]; (8) Weisfeiler-Lehman Graph Kernels (WLGK) [28]; 9) Propagation kernel (PK) [24]. For method (4),(6),(7),(8),(9) we directly cited their reported results (averaged 10-fold corss-validated error) due to unavailability of their codes; for other competing methods we run their codes with default setting and report the performance.

**Experimental setting**. We follow the experimental setting in [37] and [25] and perform 10-fold cross-validation; we report the average and standard deviation of validation accuracies across the 10 folds within the cross-validation. In the SLIM network, the structural resolution is controlled by a BFS with 3-hop neighbors, and the structural resolution is simply set to $K = 100$; the FC-layer has one hidden layer with dimension 128; the loss function is the cross-entropy loss. No drop-out or batch-normalization is used considering the size of the benchmark data. The hyper-parameters for different dataset include (1) the number of hidden units in the Autoencoder with one hidden unit with a dimension $\{d, d/2, 2d\}$; (2) the optimizer is chosen among SGD or Adagrad, with the learning rate $\{1e-2, 5e-2, 1e-3, 5e-3, 1e-4\}$; (3) local graph representation, including node distribution $\mathbf{A}^{(k)}\mathbf{X}_i$, layer-wise distribution, and weighted layer-wise summation (Sec 3.2); (4) the number of epochs, i.e., a single epoch with the best cross-validated accuracy averaged over all the 10 folds was selected. Overall, a minimal SLIM network is used in the experiments in order to test its performance.

**Structural Resolution**. In Figure 4, we examine the performance of SLIM under different choices of the structural resolution (landmark set size $K$). As can be seen, the accuracy-vs-$K$ curve has a bell-shaped structure. When $K$ is either too small (underfitting) or too large (coherent landmarks that overfit), the accuracy is low, and the best performance is typically around a median $K$ value. This validates the correctness of Theorem 1, and the usefulness of structural landmarking in improving graph classification.

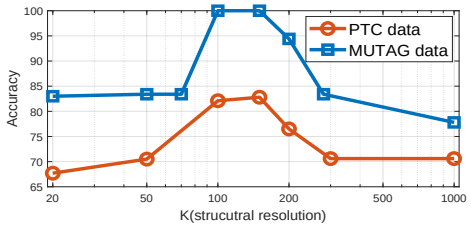
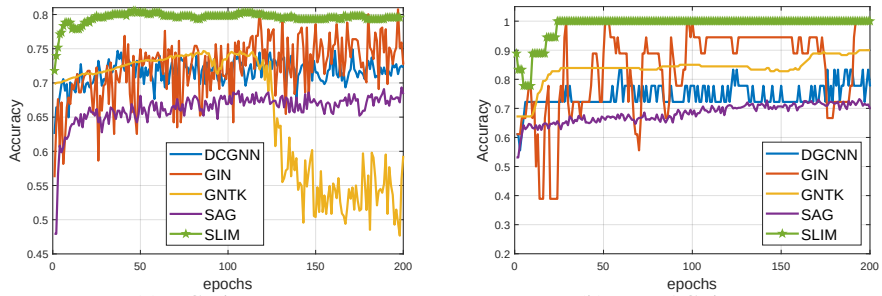
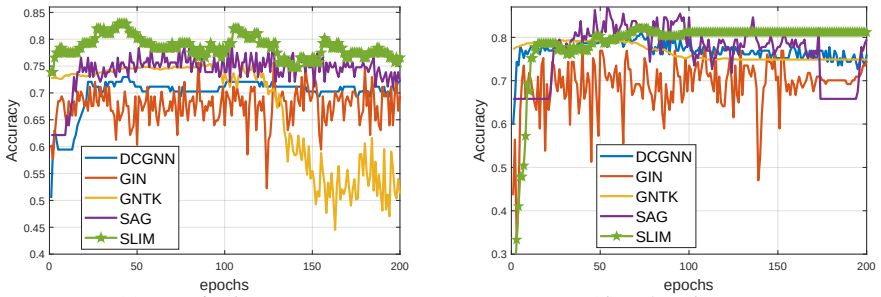


Figure 4: Accuracy vs structural resolution $K$.

**Classification Performance**. We then compare the performance of different methods in Table 1. As can be seen, overall, neural network based approaches are more competitive than graph kernels, except that graph kernels have lower fluctuations, and the WL-graph kernel perform the best on the NCI1 dataset. On most benchmamrk datasets, the SLIM network generates classification accuracies that are either higher or at least as good as other GNN/graph-pooling schemes.




(a) NCI data. (b) MUTAG data.




(c) Protein data. (d) D&D data.

Figure 5: Testing accuracy of different algorithms over the training epochs.

Table 1: Averaged prediction accuracy for different algorithms on 5 benchmark data-sets.

| Category | Algorithm | MUTAG | PTC | NCI1 | Protein | D&D |
|---|---|---|---|---|---|---|
| Graph kernel | GK | 81.38±1.74 | 55.65±0.46 | 62.49±0.27 | 71.39±0.31 | 74.38±0.69 |
| | PK | 76.00±2.69 | 59.50±2.44 | 82.54±0.47 | 73.68±0.68 | 78.25±0.51 |
| | WLGK | 84.11±1.91 | 57.97±2.49 | **84.46±0.45** | 74.68±0.49 | 78.34±0.62 |
| GNN | PATCHY-SAN | 92.63±4.21 | 60.00±4.82 | 78.59±1.89 | 75.89±2.76 | 77.12±2.41 |
| | DGCNN | 85.83±1.66 | 68.59±6.47 | 74.46±0.47 | 75.54±0.94 | 79.37±1.03 |
| | GNTK | 90.12±8.58 | 67.92±6.98 | 75.20±1.53 | 75.61±4.24 | **79.42±2.18** |
| | DiffPool | 90.52±3.98 | - | 76.53±2.23 | 75.82±3.56 | 78.95±2.40 |
| | SAG | 73.53±9.68 | 75.67±3.12 | 74.18±1.29 | 71.86±0.97 | 76.91±2.12 |
| | GIN | 90.03±8.82 | 76.25±2.83 | 79.84±4.57 | 71.28±2.65 | 77.58±2.94 |
| | SLIM | **93.28±3.36** | **80.41±6.92** | 80.53±2.01 | **77.47±4.34** | **79.48±2.66** |

**Accuracy Evolution**. We also plot the evolution of the testing accuracy for different methods on the benchmark datasets, so as to have a more comprehensive evaluation on their performance. As can be seen from Figure 5, our approach is not only more accurate on the benchmark datasets, but also the accuracy curve w.r.t. the epochs converges relatively faster and remains more stable, making it easier to determine when to stop the training process. Other algorithms, such as the GIN and SAG algorithms can also attain a high accuracy on some benchmark datasets, but the prediction performance fluctuates significantly across the training epochs, and so it could be difficult to decide when to stop training. It's also worthwhile to note that on MUTAG data the proposed method produces a classification with 100% accuracy on more than half of the runs across different folds (Figure 5(b)). It demonstrates the power of the SLIM network in capturing important graph-level features.

## 6 Conclusion

Graph neural networks represent state-of-the-art computational architecture for graph mining.In this paper, we designed the SLIM network that employs structural landmarking to resolve resolution dilemmas in graph classification and capture inherent interactions in graph-structured systems. We hope this attempt could open up possibilities in designing GNNs with informative structural priors.

# 7 Broader Impact

The proposed research studies a new architecture for graph classification, which solves an intrinsical limitation of graph neural networks, namely the resolution dilemmas. It provides new insight in graph classification and also link it to the established research of complex systems.

Our research can be applied to problems of drug discovery, where extracting relevant graph-level features for inventing and screening drug molecules with desired chemical and biological properties if of vital importance. Considering the current COVID-19 pandemic throughout the world, discovery of new drugs for curing the disease can bring significant medical, societal and economic benefits. In the meantime, our research is also applicable to other graph-structured data such as community analysis and recommendation in social networks, therefore our research can also contribute to the industry of social media in improving user experience and content recommendations.

Our research is still preliminary and needs more extensive evaluations for applications in graph-related real-world problems. Some difficulties might lead to possible societal consequences, for example, imbalance of the training data in social networks may cause potential prediction bias among different populations, which should be addressed from both algorithmic and data collection perspectives.

# References


[1] U. Alon. Network motifs: theory and experimental approaches. *Nature Reviews Genetics*, 8:450–461, 2007.

[2] J. L. Austin R. Benson, David F. Gleich. Higher-order organization of complex networks. *Science*, 353:163 – 166, 2016.

[3] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

[4] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. Geometric deep learning: Going beyond euclidean data. *IEEE Signal Processing Magazine*, pages 18–42, 2017.

[5] D. Camacho, K. Collins, R. Powers, J. Costello, and J. Collins. Next-generation machine learning for biological networks. *Cell*, 173(7):1581–1592, 2018.

[6] C. Cangea, P. Velicković, N. Jovanović, T. K. P., and Lió. Towards sparse hierarchical grap classifiers. In *preprint arXiv:1811.01287*, 2018.

[7] P. Cilliers. *Complexity and Postmodernism: Understanding Complex Systems*. Psychology Press, 1998.

[8] A. Coates and A. Y. Ng. Learning feature representations with k-means. pages 561 – 580, 1993.

[9] N. Debarsy, S. Cordier, C. Ertur, F. Nemo, D. Nourrit-Lucas, G. Poisson, and C. Vrain. *Understanding Interactions in Complex Systems, Toward a Science of Interaction*. Cambridge Scholars Publishing, 2017.

[10] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems 29*, pages 3844–3852. 2016.

[11] D. L. Donoho and X. Huo. Uncertainty principles and ideal atomic decomposition. *IEEE Trans. Inf. Theory*, 47:2845–2862, 2001.

[12] S. S. Du, K. Hou, R. R. Salakhutdinov, B. Poczos, R. Wang, and K. Xu. Graph neural tangent kernel: Fusing graph neural networks with graph kernels. In *Advances in Neural Information Processing Systems 32*, pages 5723–5733, 2019.

[13] H. Gao and S. Ji. Graph u-net. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

[14] W. L. Hamilton, R. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017.

[15] L. Hartwell, J. Hopfield, S. Leibler, and A. Murray. From molecular to modular cell biology. *Nature*, 2:47–52, Dec 1999.

[16] N. Kriegeskorte. Deep neural networks: A new framework for modeling biological vision and brain information processing. *Annual Review of Vision Science*, 1:417–446, 2015.

[17] J. Lee, I. Lee, and J. Kang. Self-attention graph pooling. In *International Conference of Machine Learning*, pages 3734–3743, 2019.

[18] J. Lee, R. Rossi, X. Kong, X. Kong, E. K. S. Kim, and A. Rao. Graph convolutional networks with motif-based attention. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 499–508, 2019.

[19] M. Marsousi, K. Abhari, P. S. Babyn, and J. Alirezaie. An adaptive approach to learn overcomplete dictionaries with efficient numbers of elements. *IEEE Transactions on Signal Processing*, 62(12):3272 – 3283, 2014.

[20] N. A. Mehta and A. G. Gray. Sparsity-based generalization bounds for predictive sparse coding. In *Proceedings of the 30th International Conference on International Conference on Machine Learning*, pages 36 – 44, 2013.

[21] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Neural and Information Processing System*, 2013.

[22] R. Milo1, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: Simple building blocks of complex networks. In *Science*, pages 824–827, 2002.

[23] C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan, and M. Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *The Thirty-third AAAI Conference on Artificial Intelligence*, 2019.

[24] M. Neumann, R. Garnett, C. Bauckhage, and K. Kersting. Propagation kernels: efficient graph kernels from propagated information. *Machine Learning*, 102(2):209–245, 2016.

[25] M. Niepert, M. Ahmed, and K. Kutzkov. Learning convolutional neural networks for graphs. In *In Proceedings of The 33rd International Conference on Machine Learning*, 2016.

[26] V. M. Ron Meir. Distortion bounds for vector quantizers with finite codebook size. *IEEE Transactions on Information Theory*, 45(5):1621 – 1648, 1999.

[27] J. Schmidt, M. R. G. Marques, S. Botti, and M. A. L. Marques. Recent advances and applications of machine learning in solid-state materials science. *NPJ Computational Material*, 5:1581–1592, 2019.

[28] N. Shervashidze, P. Schweitzer, E. J. van Leeuwen, K. Mehlhorn, and K. M. Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(77):2539–2561, 2011.

[29] N. Shervashidze, S. Vishwanathan, T. Petri, K. Mehlhorn, and K. Borgwardt. Efficient graphlet kernels for large graph comparison. In D. van Dyk and M. Welling, editors, *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, volume 5 of *Proceedings of Machine Learning Research*, pages 488–495, 16–18 Apr 2009.

[30] N. Shervashidze, S. Vishwanathan, T. Petri, K. Mehlhorn, and K. Borgwardt. Efficient graphlet kernels for large graph comparison. In *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, volume 5, pages 488–495, 2009.

[31] J. M. Stokes, K. S. Kevin Yang, W. Jin, A. Cubillos-Ruiz, N. M. Donghia, C. R. MacNair, S. French, L. A. Carfrae, Z. Bloom-Ackermann, V. M. Tran, A. Chiappino-Pepe, A. H. Badran, I. W. Andrews, E. J. Chory, G. M. Church, E. D. Brown, T. S. Jaakkola, R. Barzilay, and J. J. Collins. A deep learning approach to antibiotic discovery. *Cell*, 180(4):688–702, 2020.

[32] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2017.

[33] L. Wang, B. Zong, Q. Ma, W. Cheng, J. Ni, W. Yu, Y. Liu, D. Song, H. Chen, and Y. Fu. Inductive and unsupervised representation learning on graph structured objects. In *International COnference on Learning Representations*, 2019.

[34] S. Wernicke. Efficient detection of network motifs. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 3(4):347–359, 2006.

[35] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[36] J. Xie, R. Girshick, and A. Farhadi. Unsupervised deep embedding for clustering analysis. In *International Conference on Learning Representations*, 2016.

[37] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

[38] P. Yanardag and S. V. N. Vishwanathan. Deep graph kernels. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 1365–1374, 2015.

[39] C. Yang, M. Liu, V. W. Zheng, and J. Han. Node, motif and subgraph: Leveraging network functional blocks through structural convolution. In *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 2018.

[40] R. Ying, J. You, C. Morris, X. Ren, W. L. Hamilton, and J. Leskovec. Hierarchical graph representation learning with differentiable pooling. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, page 4805–4815, 2018.

[41] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola. Deep sets. In *Advances in Neural Information Processing Systems 30*, pages 3391–3401. 2017.

[42] M. Zhang, Z. Cui, M. Neumann, and Y. Chen. An end-to-end deep learning architecture for graph classification. In *The Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[43] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, and M. Sun. Graph neural networks: A review of methods and applications. In *ArXiv*. 2018.

# 8 Appendix

## 8.1 Proof of Theorem I

*Proof.* Suppose we have $n$ spatial instances embedded in the $d$-dimensional latent space as $\{\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_n\}$, and the landmarks (or codevectors) are defined as $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, ..., \boldsymbol{\mu}_K$. Let $p(\mathbf{z})$ be the density function of the instances. Define the averaged distance between the instance and the closest landmark point as

$$s = \frac{1}{n} \sum_{i=1}^{n} \|\mathbf{z}_i - \boldsymbol{\mu}_{c(i)}\|_2, \tag{6}$$

where $c(i)$ is the index of the closest landmark to instance $i$. As expected, $s$ will decay with the number of landmarks with the following rate [26]

$$s \leq C_d C_p \left( \left\lfloor \left( \frac{K}{2} \right)^{\frac{1}{d}} \right\rfloor^{-1} + 1 \right) K^{-\frac{1}{d}} \tag{7}$$

where $C_d$ is a dimension-dependent factor $C_d = \frac{3}{2}\left(1 + \frac{\log(d)}{d}\right)\gamma_d$, with $V_d = 2\Gamma(\frac{1}{2})^d / d\Gamma(\frac{d}{2})$ the volume of the unit ball in k-dimensional Euclidean space and $\gamma_d = 1 + d\log(d\log(d))$; $C_p = \left(\int p(\mathbf{z})^{\frac{d}{d+1}} d\mathbf{z}\right)^{\frac{d+1}{d}}$ is a factor depending on the distribution $p$.

Since $s$ is the average distortion error, we can make sure that there exists a non-empty subset of instances $\Omega_z$ such that $\|\mathbf{z}_i - \boldsymbol{\mu}_{c(i)}\| \leq s$ for $i \in \Omega_z$. Next we will only consider this subset of instances and the relevant set of landmarks will be denoted by $\Omega_u$. For the landmarks $\boldsymbol{\mu}_p \in \Omega_u$, we make a realistic assumption that there are enough instances so that we can always find one instance $\mathbf{z}$ falling in the middle of $\boldsymbol{\mu}_p$ and its closest landmark neighbor $\boldsymbol{\mu}_p$. In this case, we have then bound the distance between the closest landmark pairs as

$$\|\boldsymbol{\mu}_p - \boldsymbol{\mu}_q\| \leq \|\boldsymbol{\mu}_p - \mathbf{z}\|_2 + \|\boldsymbol{\mu}_q - \mathbf{z}\|_2 \leq 2s.$$

For any such pair, assume that the angle spanned by them is $\theta_{pq}$. we can bound the angle between the two landmark vectors by

$$\sin(\theta_{pq}) \leq \frac{2s}{\|\boldsymbol{\mu}_p\|}. \tag{8}$$

Let $u_{max} = \max_{\boldsymbol{\mu}_p \in \Omega_u} \|\boldsymbol{\mu}_p\|_2$, we can finally low-bound the normalized correlation between close landmark pairs, and henceforth the coherence of the landmarks, as

$$\begin{aligned}
\mu^2(\mathbf{U}) &\geq \max_{p,q \in \Omega_u} \cos^2(\theta_{pq}) \\
&= \max_{p,q \in \Omega_u} 1 - \sin^2(\theta_{pq})^2 \\
&\geq 1 - \frac{4s^2}{u_{max}^2} \\
&\geq 1 - \frac{4C_d K^{-\frac{1}{d}}}{u_{max}^2} \left( \left\lfloor \left( \frac{K}{2} \right)^{\frac{1}{d}} \right\rfloor^{-1} + 1 \right)
\end{aligned}$$

This indicates that the squared mutual coherence of the landmarks has a lower bound that consistently increases when the number of the landmark vectors, $K$, increases in a dictionary learning process. □

This theorem provides important guidance on the choice of structural resolution. It shows that when a clustering-based dictionary learning scheme is used to determine the structural landmarks, the size of the dictionary $K$ can not be chosen too large; or else the risk of overfitting can be huge. Note that exact sub-structure matching as is often practiced in current graph mining tasks corresponds to an extreme case where the number of landmarks, $K$, equals the number of unique sub-structures; therefore it should be avoided in practice. The structural landmarking scheme is a flexible framework to tune the number of landmarks, and to avoid overfitting.

## 8.2 Choice of Spatial and Structural Resolutions

The spatial resolution can be taken as the "size" of the local sub-structure (or sub-graph), such as the functional modules in a molecule. Small sub-structures can be very limited in terms of their representation power, while too large sub-structures can mask the right scale of the local components crucial to the learning task. An optimal spatial resolution can be data-dependent. In practice, we will restrict the size of the local sub-graphs to 3-hop BFS neighbors, considering that the "radius" of the graphs in the benchmark data-sets are usually around 5-8. We then further fine-tune the spatial resolution by assigning a non-negative weighting on the nodes residing on different layers from the central code in the local subgraph. Such weighting is shared across all the sub-graphs and can be used to adjust the importance of each layer of the BFS-based sub-graph. The weighting can be chosen as a monotonously decaying function, or optimized through learning.

The choice of structural resolution has a similar flavor in that too small or too large resolutions are neither desirable. On the other hand, it can be adjusted conveniently by tuning the landmark set size $K$ based on the validation data. In our experiments, for simplicity, we fix $K = 100$ because at a reasonable range the performance is not quite sensitive.

Finally, note that geometrically larger substructures (or sub-graphs) are characterized by higher variations among instances due to the exponential amount of configuration. Therefore, the structural resolution should also commensurate with spatial resolutions. For example, substructures constructed by 1-hop-BFS may use a smaller landmark size $K$ than those with 3-hop-neighbors. In our experiments we do not consider such dependencies yet, but will study it in our future research.

## 8.3 Comparison with Graph Kernels

Graph kernels are powerful methods to measure the similarity between graphs. The key idea is to compare the sub-structures between two graphs and compute the accumulated similarity, where the substructures can vary from random walks, paths, subgraphs, or sub-trees. Among then, paths/sub-graphs/sub-trees are deterministic sub-structures in a graph, while random walks are stochastic sequences (of nodes) in a graph.

Although SLIM network considers sub-structures as the basic processing unit, it has a number of important differences compared with graph kernels. First, we consider optimizable sub-structural landmarks, whose computation is dependent on the class labels and therefore discriminative; in comparison, the sub-structures considered in graph kernels are identified by enumerating or sampling among a large amount of pre-determined candidates. Second, the similarity measured by graph kernels is quite different from the interaction modelling as in the SLIM network. Third, it can be difficult to interpret graph kernels due to the nonlinearity of kernel methods and the exponential amount of sub-structures; in comparison, the SLIM network maintains a reasonable amount of "landmark" structures and so can provide informative clues on the prediction result.

## 8.4 Hierarchical Version

### 8.4.1 Subtlety in Spatial Resolution Definition

First we would like to clarify a subtlety in the definition of spatial resolutions. In physics, resolution is defined as the smallest distance (or interval) between two objects that can be separated; therefore it must involve two scales: the scale of the object, and the scale of the interval. Usually these two scales are proportional. In other words, you cannot have a large intervals and small objects, or the opposite (a small interval and large object). For example, in the context of imaging, each object is a pixel and the size of the pixel is the same as the interval between two adjacent pixels.

In the context of graphs, each object is a sub-graph centered around one node, whose scale is mannually determined by the order of the BFS-search centered around that node. On the other hand, the interval between two sub-graphs may be smaller than the object size. For example, two nodes $i$ and $j$ are direct neighbors, and each of them can have a 3-hop sub-graph. Then, the interval between these two subgraphs, if defined by the distance between their respective sub-graph centers, will be 1-hop; this is smaller than the size of the sub-graph, which is 3-hop. In other words, the two objects indeed overlap with each other, and the scale of the object and the scale of the interval between objects is no longer commensurate (large objects and small interval in this scenario).

This scenario makes it less complete to define spatial resolutions just based on the size of the sub-graphs (as in the main text), since there are actually two scales to define. To avoid unnecessary confusions, we skip these details. In practice, one has two choices dealing with the discrepancy: (1) requiring that the sub-graphs are not overlapping, i.e., we do not have to grow one $k$-hop subgraph around each node; instead, we just explore a subset of the sub-graphs. This can be implemented in a hierarchical version which we discuss in the next subsection; (2) we still allow each node to have a local sub-graph and study them together, which helps cover the diversity of subgraphs since theoretically, an ideal choice of the subgraph is highly domain specific and having more sub-graph examples gives a better chance to include those sub-graphs that are beneficial to the prediction task.

### 8.4.2 Hierarchical SLIM

We can implement a hierarchical version of SLIM so that sub-graphs of different scales, together with the interacting relation between sub-graphs under each scale, can be captured for final prediction. Note that in [40] a hierarchical clustering scheme is used to partition one graph, in a bottom up manner, to less and less clusters. We can implement the same idea and construct a hierarchy of scales each of which will host a number of sub-structures. The structural landmarking scheme will be implemented in each layer of the hierarchy to generate graph-level features specific to that scale. Finally these features can be combined together for graph classification.

## 8.5 Semi-supervised SLIM Network

The SLIM network is flexible and can be trained in both fully supervised setting and semi-supervised setting. This is because the SLIM model takes a parametric form and so it is inductive and can generalize to any new samples; on the other hands, the clustering-based loss term in (3) can be evaluated on both labeled samples and unlabeled samples, rendering the extra flexibility to look into the distribution of the testing sample in the training phase, if they are available. This is in flavor very similar to the smoothness constraint widely used in semi-supervised learning, such as the graph-regularized manifold learning [3]. Therefore, the SLIM network can be implemented in the following modes

- Supervised version. Only training graphs and their labels are available during the training phase, and the loss function (3) is only computed on the training samples.
- Semi-supervised version. Both labeled training graphs and unlabeled testing graphs are available. The loss function (3) will be computed on both the training and testing graphs, wile the classification loss function will only be evaluated on the training graph labels.

## 8.6 Interpretability

The SLIM network not only generates accurate prediction in graph classification problems, but can also provide important clues on interpreting the prediction results, because the graph-level features in SLIM bear clear physical meaning. For example, assume that we use the interaction matrix $\mathbf{C}_i$ for the $i$th graph $\mathcal{G}_i$ as its feature representation; and the $pq$th entry then quantifies the connectivity strength between the $p$th sub-structure landmark and the $q$th structure landmark. Then, by checking the $K^2$-dimensional model coefficients from the fully-connected layer, one can then tell which subset of substructure-connectivity (i.e., two substructures are directly connected in a graph) is important in making the prediction. To improve the interpretability one can further imposes a sparsity constraint on the model coefficient.

In traditional graph neural networks such as GraphSAGE of GIN, node features are transformed through many layers and finally mingled altogether through graph pooling. The resultant graph-level representation, whose dimension is manually determined and each entry pools the values across all the nodes in the graph, defies any effort to interpret the model.

## 8.7 The prediction Layer

The SLIM network renders various possibilities to generate the prediction layer.

- Fully connected layer. The interaction matrix can be re-shaped into a vector, or transformed to a smaller matrix via bilateral dimension reduction before reshaped into a vector. Then a fully connected layer follows for the final prediction.
- Landmark Graph. Each graph $\mathcal{G}_i$ can be transformed into a landmark-graph a with fixed number of $K$ (landmark) nodes, with $\mathbf{p}_i$ and $\mathbf{C}_i$ quantifying the weight of each node and the edge between every pair of nodes, and $M_i$ the feature of each node (see definition in Section3.3). Then, this graph can be subject to a graph convolution such as $\mathbf{D}_i^{-1}\mathbf{A}\mathbf{M}_i$ generate a fixed-dimensional graph-level feature without having to take care of the varying graph size. We will study this in our future experiments.
- Riemannian manifold. When using the interaction matrix $_i$ or the normalized version as graph level features, we can treat each graph as a point in the Riemannian manifold due to the symmetry and positive semi-definiteness of the representation. Then the distance between two interaction matrices can be computed as the Wasserstein distance between two Gaussian distributions with the interaction matrix as covariances, which has a closed-form. We will study this in our future experiments.

### 8.8 Interaction versus Integration

The SLIM network and existing GNNs represent two different flavors of learning, namely, interaction modelling versus integration approach. Interaction modelling is based on mature understanding of complex systems and can provide physically meaningful interpretations or support for graph classification; integration based approaches bypass the difficulty of preserving the identity of sub-structures and instead focus on whether the integrated representation is an injective mapping, as typically studied in graph isomorphism testing.

Note that an ideal classification is different from isomorphism testing and is not injective. In a good classifier, the goal of deciding which samples are similar and which are not are equally important. Here comes the tradeoff between handling similarity and distinctness. The Isomorphism-flavor GNN's are aimed at preserving the differences between local sub-structures (even just a very minute difference), and then map the resultant embedding to the class labels. Our approach, on the other hand, tries to absorb patterns that are sufficiently close to the same landmark, and then map the landmark-based features to class labels. In the latter case, the structural resolution can be tuned in a flexible way to explore different fineness levels, and explicitly preserving the structural landmarks allows preserving substructure identities and exploiting their interactions.