Identity-aware Graph Neural Networks

Jiaxuan You, Jonathan M Gomes-Selman, Rex Ying, Jure Leskovec
Department of Computer Science, Stanford University
{jiangxuan, gomes, rexy, jure}@cs.stanford.edu

Abstract

Message passing Graph Neural Networks (GNNs) provide a powerful modeling framework for relational data. However, the expressive power of existing GNNs is upper-bounded by the 1-Weisfeiler-Lehman (1-WL) graph isomorphism test, which means GNNs that are not able to predict node clustering coefficients and shortest path distances, and cannot differentiate between different \( d \)-regular graphs. Here we develop a class of message passing GNNs, named Identity-aware Graph Neural Networks (ID-GNNs), with greater expressive power than the 1-WL test. ID-GNN offers a minimal but powerful solution to limitations of existing GNNs. ID-GNN extends existing GNN architectures by inductively considering nodes’ identities during message passing. To embed a given node, ID-GNN first extracts the ego network centered at the node, then conducts rounds of heterogeneous message passing, where different sets of parameters are applied to the center node than to other surrounding nodes in the ego network. We further propose a simplified but faster version of ID-GNN that injects node identity information as augmented node features. Altogether, both versions of ID-GNN represent general extensions of message passing GNNs, where experiments show that transforming existing GNNs to ID-GNNs yields on average 40% accuracy improvement on challenging node, edge, and graph property prediction tasks; 3% accuracy improvement on node and graph classification benchmarks; and 15% ROC AUC improvement on real-world link prediction tasks. Additionally, ID-GNNs demonstrate improved or comparable performance over other task-specific graph networks.

Introduction

Graph Neural Networks (GNNs) represent a powerful learning paradigm that have achieved great success (Scarselli et al. 2008; Li et al. 2016; Kipf and Welling 2017; Hamilton, Ying, and Leskovec 2017; Velickovic et al. 2018; Xu et al. 2019; You, Ying, and Leskovec 2020). Among these models, messaging passing GNNs, such as GCN (Kipf and Welling 2017), GraphSAGE (Hamilton, Ying, and Leskovec 2017), and GAT (Velickovic et al. 2018), are dominantly used today due to their simplicity, efficiency and strong performance in real-world applications (Zitnik and Leskovec 2017; Ying et al. 2018; You et al. 2018a, 2019b, 2020a,b). The central idea behind message passing GNNs is to learn node embeddings via the repeated aggregation of information from local node neighborhoods using non-linear transformations (Battaglia et al. 2018).

Although GNNs represent a powerful learning paradigm, it has been shown that the expressive power of existing GNNs is upper-bounded by the 1-Weisfeiler-Lehman (1-WL) test (Xu et al. 2019). Concretely, a fundamental limitation of existing GNNs is that two nodes with different neighborhood structure can have the same computational graph, thus appearing indistinguishable. Here, a computational graph specifies the procedure to produce a node’s embedding. Such failure cases are abundant (Figure 1): in node classification tasks, existing GNNs fail to distinguish nodes that reside in \( d \)-regular graphs of different sizes; in link prediction tasks, they cannot differentiate node candidates with the same neighborhood structures but different shortest path distance to the source node; and in graph classification tasks, they cannot differentiate \( d \)-regular graphs (Chen et al. 2019; Murphy et al. 2019). While task-specific feature augmentation can be used to mitigate these failure modes, the process of discovering meaningful features for different tasks is not generic and can, for example, hamper the inductive power of GNNs.

Several recent methods aim to overcome these limitations in existing GNNs. For graph classification tasks, a collection of works propose novel architectures more expressive than the 1-WL test (Chen et al. 2019; Maron et al. 2019a; Murphy et al. 2019). For link level tasks, P-GNNs are proposed to overcome the limitation of existing GNNs (You, Ying, and Leskovec 2019). While these methods have a rich theoretical grounding, they are often task specific (either graph or link level) and often suffer from increased complexity in computation or implementation. In contrast, message passing GNNs have a track record of high predictive performance across node, link, and graph level tasks, while being simple and efficient to implement. Therefore, extending message passing GNNs beyond the expressiveness of 1-WL test, to overcome current GNN limitations, is a problem of high importance.

**Present work.** Here we propose Identity-aware Graph Neural Networks (ID-GNNs), a class of message passing GNNs with expressive power beyond the 1-WL test\(^1\). ID-GNN provides a universal extension and makes any existing message passing GNN more expressive. ID-GNN embeds each node

\(^1\)Project website with code: http://snap.stanford.edu/idgnn
by *inductively* taking into account its identity during message passing. The approach is different from labeling each node with a one-hot encoding, which is *transductive* (cannot generalize to unseen graphs). As shown in Figure 1, we use an *inductive identity coloring* technique to distinguish a node itself (the root node in the computational graph) from other nodes in its local neighborhood, within its respective computational graph. This added identity information allows ID-GNN to distinguish what would be identical computational graphs across node, edge and graph level tasks, and this way overcome the previously discussed limitations.

We propose two versions of ID-GNN. As a general approach, identity information is incorporated by applying rounds of *heterogeneous message passing*. Specifically, to embed a given node, ID-GNN first extracts the ego network centered at that node, then applies message passing, where the messages from the center node (colored nodes in Figure 1) and the rest of the nodes are computed using *different sets of parameters*. This approach naturally applies to applications involving node or edge features. We also consider a simplified version of ID-GNN, where we inject identity information via cycle counts originating from a given node as augmented node features. These cycle counts capture node identity information by counting the colored nodes within each layer of the ID-GNN computational graph, and can be efficiently computed by powers of a graph's adjacency matrix.

We compare ID-GNNs against GNNs across 8 datasets and 6 different tasks. First, we consider a collection of challenging graph property prediction tasks where existing GNNs fail, including predicting node clustering coefficient, predicting shortest path distance, and differentiating random $d$-regular graphs. Then, we further apply ID-GNNs to real-world datasets. Results show that transforming existing GNNs to their ID-GNN versions yields on average 40% accuracy improvement on challenging node, edge, and graph property prediction tasks; 3% accuracy improvement on node and graph classification benchmarks; and 15% ROC AUC improvement on real-world link prediction tasks. Additionally, we compare ID-GNNs against other expressive graph networks that are specifically designed for edge or graph-level tasks. ID-GNNs demonstrate improved or comparable performance over these models, further emphasizing the versatility of ID-GNNs.

Our key contribution includes: (1) We show that message passing GNNs can have expressive power beyond 1-WL test. (2) We propose ID-GNNs as a general solution to the limitations in existing GNNs, with rich theoretical and experimental results. (3) We present synthetic and real-world tasks to reveal the failure modes of existing GNNs and demonstrate the superior performance of ID-GNNs over both existing GNNs and other powerful graph networks.

**Related Work**

**Expressive neural networks beyond 1-WL test.** Recently, many neural networks have been proposed with expressive power beyond the 1-WL test, including (Chen et al. 2019; Maron et al. 2019a; Murphy et al. 2019; You, Ying, and Leskovec 2019; Li et al. 2020). However, these papers introduce extra, often task/domain specific, components beyond standard message passing GNNs. For example, P-GNN’s embeddings are tied with random anchor-sets and, thus, are not applicable to node/graph level tasks which require deter-
We design ID-GNN so that it can make any message passing GNN more expressive. ID-GNN is built with two important components: (1) inductive identity coloring where identity information is injected to each node, and (2) heterogeneous message passing where the identity information is utilized in message passing. Algorithm 1 provides an overview.

**Inductive identity coloring.** To embed a given node \( v \in \mathcal{G} \) using a \( K \)-layer ID-GNN, we first extract the \( K \)-hop ego network \( \mathcal{G}^{(K)}_v \) of \( v \). We then assign a unique coloring to the central node of the ego network \( \mathcal{G}^{(K)}_v \). Altogether, nodes in \( \mathcal{G}^{(K)}_v \) can be categorized into two types throughout the embedding process: nodes with coloring and nodes without coloring. This coloring technique is inductive because even if nodes are permuted, the center node of the ego network can still be differentiated from other neighboring nodes.

**Heterogeneous message passing** \( K \) rounds of message passing are then applied to all the extracted ego networks. To embed node \( u \in \mathcal{G}^{(K)}_v \), we extend Eq. 1 to enable heterogeneous message passing:

\[
\begin{align*}
\mathbf{m}_u^{(k)} &= \text{MSG}^{(k)}(\mathbf{m}_u^{(k-1)}), \\
\mathbf{h}_u^{(k)} &= \text{AGG}^{(k)}(\{\mathbf{m}_u^{(k)}, s \in \mathcal{N}(u)\}, \mathbf{h}_u^{(k-1)})
\end{align*}
\]

where only \( \mathbf{h}_u^{(k)} \) is used as the embedding representation for node \( v \) after applying \( K \) rounds of Eq. 3. Different from Eq. 1, two sets of MSG\((k)\) functions are used, where MSG\((k)\) is applied to nodes with identity coloring, and MSG\((0)\) is used for node without coloring. The indicator function \( 1[s = v] = 1 \) if \( s = v \) else 0 is used to index the selection of these functions. This way, the inductive identity coloring is encoded into the ID-GNN computational graph.

A benefit of this heterogeneous message passing approach is that it is applicable to any message passing GNN. For example, consider the following message passing scheme, which extends the definition of GNNs in Eq. 3 by including edge attributes \( f_{uv} \) during message passing:

\[
\begin{align*}
\mathbf{m}_{su}^{(k)} &= \text{MSG}^{(k)}_{1[sv]}(\mathbf{m}_u^{(k-1)}, f_{su}), \\
\mathbf{h}_u^{(k)} &= \text{AGG}^{(k)}(\{\mathbf{m}_u^{(k)}, s \in \mathcal{N}(u)\}, \mathbf{h}_u^{(k-1)})
\end{align*}
\]
one, the computation of ID-GNN is almost identical to the widely used mini-batch version of GNNs (Hamilton, Ying, and Leskovec 2017; Ying et al. 2018). In our experiments, by matching the number of trainable parameters, the computation FLOPS used by ID-GNNs and mini-batch GNNs can be the same (shown in Table 4).

**Extension to edge-level tasks.** Here we discuss how to extend the ID-GNN framework to properly resolve existing GNN limitations in edge-level tasks (Figure 1, middle). Suppose we want to predict the edge-level label for a node pair \( u, v \). For ID-GNN, the prediction is made from a conditional node embedding \( \mathbf{h}_{u[v]} \), which is computed by assigning node \( v \), rather than \( u \), identity coloring in node \( u \)'s computation graph, as illustrated in Figure 1. In the case where node \( v \) does not lie within \( u \)'s K-hop ego network, no identity coloring is used and ID-GNNs will still suffer from existing failure cases of GNNs. Therefore, we use deeper ID-GNNs for edge-level prediction tasks in practice.

**ID-GNNs Expressive Power: Theoretical Results**

**ID-GNNs are strictly more expressive than existing message passing GNNs.** It has been shown that existing message passing GNNs have an expressive power upper bound by the 1-WL test, where the upper bound can be instantiated by the widely used mini-batch version of GNNs (Hamilton, Ying, and Leskovec 2017; Ying et al. 2018). In our experiments, by matching the number of trainable parameters, the computation FLOPS used by ID-GNNs and mini-batch GNNs can be the same (shown in Table 4).

**Proposition 1.** ID-GNN version of GIN can differentiate any graph that GIN can differentiate, while being able to differentiate certain graphs that GIN fails to distinguish.

By setting \( \text{Msg}_G^{(k)}(\cdot) = \text{Msg}_I^{(k)}(\cdot) \), Eq. 3 becomes identical to Eq. 1 which trivially proves the first part. The \( d \)-regular graph example given in Figure 1 then proves the second part.

**ID-GNNs can count cycles.** Proposition 1 provides an overview of the added expressive power of ID-GNNs. Here, we reveal one concrete aspect of this added expressive power, i.e., ID-GNN’s capability to count cycles. We observe that the ability of counting cycles is intuitive to understand; moreover, it is crucial for useful tasks such as predicting node clustering coefficient, which we elaborate in the next section.

**Proposition 2.** For any node \( v \), there exists a \( K \)-layer ID-GNN instantiation that can learn an embedding \( \mathbf{h}_v^{(K)} \) where the \( j \)-th dimension \( h_v^{(K)}[j] \) equals the number of length \( j \) cycles starting and ending at node \( v \). We prove this by showing that ID-GNNs can count paths from any node \( u \) to the identity node \( v \). Through induction, we show that a 1-layer ID-GNN embedding \( \mathbf{h}_u^{(1)} \) can count length 1 paths from \( u \) to \( v \). Then, given a \( K \)-layer ID-GNN embedding \( \mathbf{h}_u^{(K)} \) that counts paths of length \( 1, \ldots, K \) between \( u \) and \( v \), we show the \( K + 1 \)-th layer of ID-GNN can accurately update \( \mathbf{h}_u^{(K+1)} \) to account for paths of length \( K + 1 \). Detailed proofs are provided in the Appendix.

**ID-GNNs Expressive Power: Case Studies**

**Node-level: Predicting clustering coefficient.** Here we show that existing message passing GNNs fail to inductively predict clustering coefficients purely from graph structure, while ID-GNNs can. Clustering coefficient is a widely used metric that characterizes the proportion of closed triangles in a node’s 1-hop neighborhood (Watts and Strogatz 1998). The node classification failure case in Figure 1 demonstrates GNNs’ inability to predict clustering coefficients, as GNNs fail to differentiate nodes \( v_1 \) and \( v_2 \) with clustering coefficient 1 and 0 respectively. By using one-hot node features, GNNs can overcome this failure mode (Hamilton, Ying, and Leskovec 2017). However, in this case GNNs are memorizing the clustering coefficients for each node, since one-hot encodings prevent generalization to unseen graphs.

Based on Proposition 2, ID-GNNs can learn node embeddings \( \mathbf{h}_v^{(K)} \), where \( h_v^{(K)}[j] \) equals the number of length \( j \) cycles starting and ending at node \( v \). Given these cycle counts, we can then calculate clustering coefficient \( c_u \) of node \( u \):

\[
c_v = \frac{\left| \{ e_{su} : s, u \in \mathcal{N}(v), e_{su} \in \mathcal{E} \} \right|}{(d_v)(d_v - 1)/2}
\]

\[
= \frac{\mathbf{h}_v^{(K)}[3]}{\mathbf{h}_v^{(K)}[2] * (\mathbf{h}_v^{(K)}[2] - 1)}
\]

where \( d_v \) is the degree of node \( v \). Since \( c_v \) is a continuous function of \( \mathbf{h}_v^{(K)} \), we can approximate it to an arbitrary precision with an MLP due to the universal approximation theorem (Hornik et al. 1989).

**Edge-level: Predicting reachability or shortest path distance.** Vanilla GNNs make edge-level predictions from pairs of node embeddings (Hamilton, Ying, and Leskovec 2017). However, this type of approaches fail to predict reachability or shortest path distance (SPD) between node pairs. For example, two nodes can have the same GNN node embedding, independent of whether they are located in the same connected component. Although (Veličković et al. 2020) shows that proper node feature initialization allows for the prediction of reachability and SPD, ID-GNNs present a general solution to this limitation through the use of conditional node embeddings. As discussed in “Extension to edge-level tasks”, we re-formulate edge-level prediction as conditional node-level prediction; consequently, a \( K \)-layer ID-GNN can predict if node \( u \in \mathcal{G} \) is reachable from \( v \in \mathcal{G} \) within \( K \) hops by using the conditional node embedding \( \mathbf{h}_{u|v}^{(K)} \) via:

\[
m_{s|v}^{(k)} = \begin{cases} 1 & \text{if } 1[s = v] = 1 \\ \mathbf{h}_{s|v}^{(k-1)} & \text{else} \end{cases},
\]

\[
\mathbf{h}_{u|v}^{(k)} = \text{MAX} \left\{ \{ m_{s|v}^{(k)}, s \in \mathcal{N}(u) \} \right\}
\]

where \( \mathbf{h}_{u|v}^{(0)} = 0, \forall u \in \mathcal{G} \), and the output \( \mathbf{h}_{u|v}^{(K)} = 1 \) if an ID-GNN predicts \( u \) are reachable from \( v \).

**Graph-level: Differentiating random \( d \)-regular graphs.** As is illustrated in Figure 1, existing message passing GNNs cannot differentiate random \( d \)-regular graphs purely from graph structure, as the computation graphs for each node are identical, regardless of the number of layers. Here, we show that ID-GNNs can differentiate a significant proportion of random \( d \)-regular graphs. Specifically, we generate 100 non-isomorphic random \( d \)-regular graphs and consider 3 settings with different graph sizes \( n \) and node degree \( d \). We use up to length \( K \) cycle counts, which a \( K \)-layer ID-GNN can
1-WL test or GNNs regardless of the number of layers.

We bin over continuous clustering coefficients to make task (1) challenging graph property prediction tasks: (2) BZR (Holme and Kim 2002) and (3) scale molecule dataset and (4) PROTEINS classifying nodes by clustering coefficients, (5) Strogatz 1998), each containing 256 graphs, with average clustering coefficient in the range 0.01. Due to the different nature of these tasks, tasks (1)(3)(6) provide splits, while for all the other tasks, we use a random 80/20% train/val split and average results over 3 random splits. Validation accuracy (multi-way classification) or ROC AUC (binary classification) in the final epoch is reported.

**Models.** We present a standardized framework for fairly comparing ID-GNNs with existing GNNs. We use 4 widely adopted GNN models as base models: GAT (Velickovic et al. 2018), GCN (Kipf and Welling 2017), GIN (Xu et al. 2019), and GraphSAGE (Hamilton, Ying, and Leskovec 2017). We then transform each GNN model to its ID-GNN variants, ID-GNN-Full (based on heterogeneous message passing) and ID-GNN-Fast, holding all the other hyperparameters fixed. To further ensure fairness, we adjust layer widths, so that all the models match the number of trainable parameters of a standard GCN model (i.e., match computational budget). In summary, we run 12 models for each experimental setup, including 4 types of GNN architectures, each with 3 versions.

We use 3-layer GNNs for node and graph level tasks, and 5-layer GNNs for edge level tasks, where GCNs with 256-dim hidden units are used to set the computational budget for all 12 model variants. For ID-GNNs-Full, each layer has 2 sets of weights, thus each layer has fewer number of hidden units; for ID-GNNs-Fast, 10-dim augmented cycle counts features are used. We use ReLU activation and Batch Normalization for all the models. We use Adam optimizer with learning rate 0.01. Due to the different nature of these tasks, tasks (1)(3)(6) are excluded the ogbg-molhiv dataset, are trained for 1000 epochs, while the rest are trained for 100 epochs. For node-level tasks, GNN / ID-GNN node embeddings are directly used for prediction; for edge-level tasks, ID-GNNs-Full make predictions with conditional node embeddings, while GNNs and ID-GNNs-Fast make predictions by concatenating pairs of node embeddings and then passing the result through a 256-dim MLP; for graph-level tasks, predictions are based on a global sum pooling over node embeddings.

Overall, these comprehensive and consistent experimental settings reveal the general improvement of ID-GNNs compared with existing GNNs.

**Graph Property Prediction Tasks**

Node clustering coefficient prediction. In Table 2 we observe that across all models and datasets, both ID-GNN formulations perform at the level of or significantly outperform GNN counterparts, with an average absolute performance gain of 36.8% between the best ID-GNN and best GNN. In each dataset, both ID-GNN methods perform with near 100% accuracy for at least one GNN architecture. ID-GNN-Fast shows the most consistent improvements across models with greatest improvement in GraphSAGE. These results align with the previous discussion of using cycle counts alone to learn clustering coefficients. We defer discussion until later on ID-GNN-Full sometimes showing minimal improvement, to present a general understanding of this behavior.

<table>
<thead>
<tr>
<th>ID-GNNs</th>
<th>Layer=3</th>
<th>Layer=4</th>
<th>Layer=5</th>
<th>Layer=6</th>
</tr>
</thead>
<tbody>
<tr>
<td>n=64, d=4, 100 graphs</td>
<td>11%</td>
<td>64%</td>
<td>94%</td>
<td>100%</td>
</tr>
<tr>
<td>n=40, d=5, 100 graphs</td>
<td>14%</td>
<td>82%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>n=96, d=6, 100 graphs</td>
<td>21%</td>
<td>88%</td>
<td>100%</td>
<td>100%</td>
</tr>
</tbody>
</table>

Table 1: Percentage of random d-regular graphs that ID-GNNs can differentiate (unique graph representations / total graphs). Note, none of the graphs can be differentiated by I-WL test or GNNs regardless of the number of layers.

Experiments

Datasets. We perform experiments over 8 different datasets. We consider the synthetic graph datasets (1) ScaleFree (Holme and Kim 2002) and (2) SmallWorld (Watts and Strogatz 1998), each containing 256 graphs, with average degree of 4 and average clustering coefficient in the range [0, 0.5]. For real-world datasets we explore 3 protein datasets: (3) ENZYMES (Borgwardt et al. 2005) with 600 graphs, (4) PROTEINS (Schomburg et al. 2004) with 1113 graphs, and (5) BZR (Sutherland, O’Brien, and Weaver 2003) with 405 graphs. We also consider citation networks including (6) Cora and (7) CiteSeer (Sen et al. 2008), and a large-scale molecule dataset (8) ogbg-molhiv (Hu et al. 2020) with 41K graphs.

Tasks. We evaluate ID-GNNs over two task categories. First, we consider challenging graph property prediction tasks: (1) classifying nodes by clustering coefficients, (2) classifying pairs of nodes by their shortest path distances, and (3) classifying random graphs by their average clustering coefficients. We bin over continuous clustering coefficients to make task (1) and (3) 10-way classification tasks and threshold the shortest path distance to make task (2) a 5-way classification task. We also consider more common tasks with real-world labels, including (4) node classification, (5) link prediction, and (6) graph classification. For the ogbg-molhiv dataset we use provided splits, while for all the other tasks, we use a random 80/20% train/val split and average results over 3 random splits. Validation accuracy (multi-way classification) or ROC AUC (binary classification) in the final epoch is reported.

**Experimental Setup**

**Datasets.** We perform experiments over 8 different datasets. We consider the synthetic graph datasets (1) ScaleFree (Holme and Kim 2002) and (2) SmallWorld (Watts and Strogatz 1998), each containing 256 graphs, with average degree of 4 and average clustering coefficient in the range [0, 0.5]. For real-world datasets we explore 3 protein datasets: (3) ENZYMES (Borgwardt et al. 2005) with 600 graphs, (4) PROTEINS (Schomburg et al. 2004) with 1113 graphs, and (5) BZR (Sutherland, O’Brien, and Weaver 2003) with 405 graphs. We also consider citation networks including (6) Cora and (7) CiteSeer (Sen et al. 2008), and a large-scale molecule dataset (8) ogbg-molhiv (Hu et al. 2020) with 41K graphs.

**Tasks.** We evaluate ID-GNNs over two task categories. First, we consider challenging graph property prediction tasks: (1) classifying nodes by clustering coefficients, (2) classifying pairs of nodes by their shortest path distances, and (3) classifying random graphs by their average clustering coefficients. We bin over continuous clustering coefficients to make task (1) and (3) 10-way classification tasks and threshold the shortest path distance to make task (2) a 5-way classification task. We also consider more common tasks with real-world labels, including (4) node classification, (5) link prediction, and (6) graph classification. For the ogbg-molhiv dataset we use provided splits, while for all the other tasks, we use a random 80/20% train/val split and average results over 3 random splits. Validation accuracy (multi-way classification) or ROC AUC (binary classification) in the final epoch is reported.

**Models.** We present a standardized framework for fairly comparing ID-GNNs with existing GNNs. We use 4 widely adopted GNN models as base models: GAT (Velickovic et al. 2018), GCN (Kipf and Welling 2017), GIN (Xu et al. 2019), and GraphSAGE (Hamilton, Ying, and Leskovec 2017). We then transform each GNN model to its ID-GNN variants, ID-GNN-Full (based on heterogeneous message passing) and ID-GNN-Fast, holding all the other hyperparameters fixed. To further ensure fairness, we adjust layer widths, so that all the models match the number of trainable parameters of a standard GCN model (i.e., match computational budget). In summary, we run 12 models for each experimental setup, including 4 types of GNN architectures, each with 3 versions.

We use 3-layer GNNs for node and graph level tasks, and 5-layer GNNs for edge level tasks, where GCNs with 256-dim hidden units are used to set the computational budget for all 12 model variants. For ID-GNNs-Full, each layer has 2 sets of weights, thus each layer has fewer number of hidden units; for ID-GNNs-Fast, 10-dim augmented cycle counts features are used. We use ReLU activation and Batch Normalization for all the models. We use Adam optimizer with learning rate 0.01. Due to the different nature of these tasks, tasks (1)(3)(6) are excluded from the ogbg-molhiv dataset, are trained for 1000 epochs, while the rest are trained for 100 epochs. For node-level tasks, GNN / ID-GNN node embeddings are directly used for prediction; for edge-level tasks, ID-GNNs-Full make predictions with conditional node embeddings, while GNNs and ID-GNNs-Fast make predictions by concatenating pairs of node embeddings and then passing the result through a 256-dim MLP; for graph-level tasks, predictions are based on a global sum pooling over node embeddings.

Overall, these comprehensive and consistent experimental settings reveal the general improvement of ID-GNNs compared with existing GNNs.

**Graph Property Prediction Tasks**

Node clustering coefficient prediction. In Table 2 we observe that across all models and datasets, both ID-GNN formulations perform at the level of or significantly outperform GNN counterparts, with an average absolute performance gain of 36.8% between the best ID-GNN and best GNN. In each dataset, both ID-GNN methods perform with near 100% accuracy for at least one GNN architecture. ID-GNN-Fast shows the most consistent improvements across models with greatest improvement in GraphSAGE. These results align with the previous discussion of using cycle counts alone to learn clustering coefficients. We defer discussion until later on ID-GNN-Full sometimes showing minimal improvement, to present a general understanding of this behavior.
Table 2: Comparing ID-GNNs with GNNs on graph property prediction tasks. For each column, all the 12 models have the same computational budget. The best performance in each family of models is bold. Results are averaged over 3 random splits.

<table>
<thead>
<tr>
<th>GNNs</th>
<th>ScaleFree</th>
<th>SmallWorld</th>
<th>ENZYMES</th>
<th>PROTEINS</th>
<th>ScaleFree</th>
<th>SmallWorld</th>
<th>ENZYMES</th>
<th>PROTEINS</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCN</td>
<td>0.679 ± 0.01</td>
<td>0.589 ± 0.04</td>
<td>0.596 ± 0.02</td>
<td>0.540 ± 0.00</td>
<td>0.522 ± 0.00</td>
<td>0.558 ± 0.02</td>
<td>0.557 ± 0.02</td>
<td>0.722 ± 0.01</td>
</tr>
<tr>
<td>SAGE</td>
<td>0.470 ± 0.03</td>
<td>0.271 ± 0.03</td>
<td>0.572 ± 0.04</td>
<td>0.444 ± 0.03</td>
<td>0.297 ± 0.01</td>
<td>0.360 ± 0.16</td>
<td>0.580 ± 0.02</td>
<td>0.722 ± 0.01</td>
</tr>
<tr>
<td>GAT</td>
<td>0.470 ± 0.03</td>
<td>0.274 ± 0.06</td>
<td>0.464 ± 0.03</td>
<td>0.400 ± 0.02</td>
<td>0.451 ± 0.00</td>
<td>0.551 ± 0.02</td>
<td>0.556 ± 0.02</td>
<td>0.722 ± 0.01</td>
</tr>
<tr>
<td>GIN</td>
<td>0.693 ± 0.00</td>
<td>0.571 ± 0.04</td>
<td>0.660 ± 0.02</td>
<td>0.558 ± 0.02</td>
<td>0.551 ± 0.01</td>
<td>0.575 ± 0.02</td>
<td>0.541 ± 0.03</td>
<td>0.722 ± 0.01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID-GNNS Fast</th>
<th>ScaleFree</th>
<th>SmallWorld</th>
<th>ENZYMES</th>
<th>PROTEINS</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCN</td>
<td>0.897 ± 0.01</td>
<td>0.812 ± 0.02</td>
<td>0.786 ± 0.04</td>
<td>0.805 ± 0.02</td>
</tr>
<tr>
<td>SAGE</td>
<td>0.954 ± 0.01</td>
<td>0.994 ± 0.00</td>
<td>0.958 ± 0.04</td>
<td>0.985 ± 0.01</td>
</tr>
<tr>
<td>GAT</td>
<td>0.889 ± 0.01</td>
<td>0.739 ± 0.03</td>
<td>0.675 ± 0.04</td>
<td>0.675 ± 0.05</td>
</tr>
<tr>
<td>GIN</td>
<td>0.895 ± 0.00</td>
<td>0.822 ± 0.03</td>
<td>0.798 ± 0.06</td>
<td>0.790 ± 0.01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID-GNNS Full</th>
<th>ScaleFree</th>
<th>SmallWorld</th>
<th>ENZYMES</th>
<th>PROTEINS</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCN</td>
<td>0.985 ± 0.01</td>
<td>0.994 ± 0.00</td>
<td>0.984 ± 0.02</td>
<td>0.995 ± 0.00</td>
</tr>
<tr>
<td>SAGE</td>
<td>0.958 ± 0.00</td>
<td>0.400 ± 0.12</td>
<td>0.591 ± 0.04</td>
<td>0.474 ± 0.03</td>
</tr>
<tr>
<td>GAT</td>
<td>0.638 ± 0.01</td>
<td>0.847 ± 0.20</td>
<td>0.594 ± 0.00</td>
<td>0.994 ± 0.00</td>
</tr>
<tr>
<td>GIN</td>
<td>0.716 ± 0.01</td>
<td>0.572 ± 0.04</td>
<td>0.655 ± 0.02</td>
<td>0.570 ± 0.03</td>
</tr>
</tbody>
</table>

Best ID-GNN over best GNN | 29.3% | 40.6% | 33.4% | 43.7% | 44.9% | 42.3% | 44.3% | 27.8% | 35.0% | 42.3%

Table 3: Comparing GNNs with ID-GNNs on real-world prediction tasks. For each column, all the 12 models have the same computational budget. The best performance in each family of models is bold. Results are averaged over 3 random splits.

<table>
<thead>
<tr>
<th>GNNs</th>
<th>ScaleFree</th>
<th>SmallWorld</th>
<th>ENZYMES</th>
<th>PROTEINS</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCN</td>
<td>0.848 ± 0.01</td>
<td>0.709 ± 0.01</td>
<td>0.796 ± 0.01</td>
<td>0.709 ± 0.00</td>
</tr>
<tr>
<td>SAGE</td>
<td>0.868 ± 0.01</td>
<td>0.726 ± 0.01</td>
<td>0.541 ± 0.00</td>
<td>0.512 ± 0.00</td>
</tr>
<tr>
<td>GAT</td>
<td>0.857 ± 0.01</td>
<td>0.716 ± 0.01</td>
<td>0.500 ± 0.00</td>
<td>0.500 ± 0.00</td>
</tr>
<tr>
<td>GIN</td>
<td>0.858 ± 0.01</td>
<td>0.719 ± 0.01</td>
<td>0.687 ± 0.01</td>
<td>0.722 ± 0.01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID-GNNS Fast</th>
<th>ScaleFree</th>
<th>SmallWorld</th>
<th>ENZYMES</th>
<th>PROTEINS</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCN</td>
<td>0.851 ± 0.02</td>
<td>0.715 ± 0.00</td>
<td>0.856 ± 0.03</td>
<td>0.719 ± 0.00</td>
</tr>
<tr>
<td>SAGE</td>
<td>0.866 ± 0.02</td>
<td>0.742 ± 0.01</td>
<td>0.899 ± 0.01</td>
<td>0.743 ± 0.02</td>
</tr>
<tr>
<td>GAT</td>
<td>0.864 ± 0.01</td>
<td>0.719 ± 0.01</td>
<td>0.837 ± 0.01</td>
<td>0.759 ± 0.00</td>
</tr>
<tr>
<td>GIN</td>
<td>0.863 ± 0.01</td>
<td>0.719 ± 0.01</td>
<td>0.771 ± 0.04</td>
<td>0.798 ± 0.03</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID-GNNS Full</th>
<th>ScaleFree</th>
<th>SmallWorld</th>
<th>ENZYMES</th>
<th>PROTEINS</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCN</td>
<td>0.875 ± 0.01</td>
<td>0.730 ± 0.02</td>
<td>0.841 ± 0.01</td>
<td>0.724 ± 0.03</td>
</tr>
<tr>
<td>SAGE</td>
<td>0.878 ± 0.02</td>
<td>0.729 ± 0.01</td>
<td>0.741 ± 0.01</td>
<td>0.767 ± 0.01</td>
</tr>
<tr>
<td>GAT</td>
<td>0.876 ± 0.01</td>
<td>0.729 ± 0.01</td>
<td>0.741 ± 0.01</td>
<td>0.767 ± 0.01</td>
</tr>
<tr>
<td>GIN</td>
<td>0.851 ± 0.00</td>
<td>0.725 ± 0.01</td>
<td>0.815 ± 0.01</td>
<td>0.810 ± 0.03</td>
</tr>
</tbody>
</table>

Best ID-GNN over best GNN | 1.0% | 1.6% | 9.6% | 8.7% | 19.2% | 21.9% | 8.3% | 1.8% | 2.5% | 2.0%
GraphSAGE model. In PROTEIN and BZR, ID-GNN-Full shows improvements in terms of expressive power compared to the baseline GNN model (except GIN on BZR), with greatest performance increases in GCN and GAT (avg. 3.6% and 3.0% respectively).

**Computational Cost Analysis**

We compare the runtime complexity (excluding mini-batch loading time) of ID-GNNs vs. existing GNNs, where we hold the computational budget constant across all models. Table 4 reveals that when considering the forward and backward pass, ID-GNN-Full runs 3.8x slower than its GNN equivalent but has an equivalent runtime complexity to the mini-batch implementation of GNN, while ID-GNN-Fast runs with essentially zero overhead over existing GNN implementations.

**Summary of Comparisons with GNNs**

Overall, ID-GNN-Full and ID-GNN-Fast demonstrate significant improvements over their message passing GNN counterparts, of the same computational budget, on a variety of tasks. In all tasks, the best ID-GNNs outperform the best GNNs; moreover, out of 160 model-task combinations, ID-GNNs fail to improve accuracy in fewer than 10 cases. For the rare cases where there is no improvement from ID-GNN-Full, we suspect that the model underfits since we control the complexity of models: given that ID-GNN-Full has two sets of weights (heterogeneous message passing), fewer weights are used for each message passing. For verification, if we double the computational budget, we observe that ID-GNN versions again outperform GNN counterparts.

**Comparisons with Expressive Graph Networks**

We provide additional experimental comparisons against other expressive graph networks in both edge and graph-level tasks. For edge-level task, we further compare with P-GNN (You, Ying, and Leskovec 2019) over the ENZYMES and PROTEINS datasets using the protocol introduced previously. For graph-level comparison, we include experimental results over 3 datasets: MUTAG with 182 graphs (Debnath et al. 1991), PTC with 344 graphs (Helma and Kramer 2003), and PROTEINS. We follow PPGN’s (Maron et al. 2019a) 10-fold 90/10 data splits and compare against 5 other expressive graph networks. We report numbers in the corresponding papers, and report the best ID-GNNs out of the 4 variants. **Link prediction**. We compare against P-GNNs on 2 link prediction datasets. As shown in the Table 5, we observe significant improvements using ID-GNNs compared to both its GNN counterpart and P-GNNs. These results both demonstrate ID-GNNs’ competitive performance as a general graph learning method against a task-specific model, while also highlighting ID-GNN’s improved ability to incorporate node-features compared with P-GNNs. **Graph classification**. We compare ID-GNNs against several other more powerful graph networks in the task of graph classification. Table 6 demonstrates the strong performance of ID-GNNs. ID-GNNs outperform other graph networks on the MUTAG and PROTEINS datasets; Although ID-GNNs performance then drops on the PTC dataset, they are still comparable to two out of the four powerful graph models. These strong results further demonstrate the ability of ID-GNN to outperform not only message passing GNNs, but also other powerful, task specific graph networks across a range of tasks.

**Conclusion**

We have proposed ID-GNNs as a general and powerful extension to existing GNNs with rich theoretical and experimental results. Specifically, ID-GNNs have expressive power beyond the 1-WL test. When runtime efficiency is the primary concern, we also present a feature augmented version of ID-GNN that maintains theoretical guarantees and empirical success of heterogeneous message passing, while only requiring one-time feature pre-processing. We recommend that this cycle-count feature augmentation be the new go-to node feature initialization when additional node attributes are not available. Additionally, as direct extensions to message passing GNNs, ID-GNNs can easily implemented and extended via existing code platform. Overall, ID-GNNs outperform corresponding message passing GNNs, while both maintaining the attractive properties of message passing GNNs and demonstrating competitive performance compared with other powerful/expressive graph networks. We hope ID-GNNs added expressive power and proven practical applicability can enable exciting new applications and further development of message passing GNNs.
Acknowlegments
We gratefully acknowledge the support of DARPA under Nos. FA865018C7880 (ASED), N660011924033 (MCS); ARO under Nos. W911NF-16-1-0342 (MURI), W911NF-16-1-0171 (DURIP); NSF under Nos. OAC-1835598 (CINES), OAC-1934578 (HDR), CCF-1918940 (Expeditions), IIS-2030477 (RAPID); Stanford Data Science Initiative, Wu Tsai Neurosciences Institute, Chan Zuckerberg Biohub, Amazon, Boeing, JPMorgan Chase, Docomo, Hitachi, JD.com, KDDI, NVIDIA, Dell. J. L. is a Chan Zuckerberg Biohub investigator. Jiaxuan You is supported by JPMorgan Chase PhD Fellowship and Baidu Scholarship. Rex Ying is supported by Baidu Scholarship.

Ethics Statement
GNNs represent a promising family of models for analyzing and understanding relational data. A broad range of application domains, such as network fraud detection (Akoglu, Chandy, and Faloutsos 2013; Kumar, Cheng, and Leskovec 2017; Akoglu and Faloutsos 2013), molecular drug structure discovery (You et al. 2018a,b; Jin, Barzilay, and Jaakkola 2018), recommender systems (Ying et al. 2018; You et al. 2019a), and network analysis (Kumar, Cheng, and Leskovec 2017; Morris et al. 2019; Fan et al. 2019; Ying et al. 2019) stand to be greatly impacted by the use and development of GNNs. As a direct extension of existing message passing GNNs, ID-GNNs represent a simple but powerful transformation to GNNs that re-frames the discussion on GNN expressive power and thus their performance in impactful problem domains. In comparison to other models that have expressive power beyond 1-WL tests, ID-GNNs are easy to implement with existing graph learning packages; therefore, ID-GNNs can be easily used as extensions of existing GNN models for tackling important real-world tasks, as well as themselves extended and further explored in the research space.

The simplicity of ID-GNNs presents great promise for further exploration into the expressiveness of GNNs. In particular, we believe that our work motivates further research into heterogeneous message passing and coloring schemes, as well as generic, but powerful forms of feature augmentation. By further increasing the expressiveness of message passing GNNs, we hope to enable new, important tasks to be solved across a wide range of disciplines or significant improvement on previously defined and widely adopted GNN models. Through ease of use and strong preliminary results, we believe that our work opens the doors for new explorations into the study of graphs and graph based tasks, with the potential for great improvement in existing GNN models.

References


