ABSTRACT
Graph Neural Networks (GNNs) have achieved state-of-the-art performance in many high-impact applications such as fraud detection, information retrieval, and recommender systems due to their powerful representation learning capabilities. Some nascent efforts have been concentrated on simplifying the structures of GNN models, in order to reduce the computational complexity. However, the dynamic nature of these applications requires GNN structures to be evolving over time, which has been largely overlooked so far. To bridge this gap, in this paper, we propose a simplified and dynamic graph neural network model, called SDG. It is efficient, effective, and provides interpretable predictions. In particular, in SDG, we replace the traditional message-passing mechanism of GNNs with the designed dynamic propagation scheme based on the personalized PageRank tracking process. We conduct extensive experiments and ablation studies to demonstrate the effectiveness and efficiency of our proposed SDG. We also design a case study on fake news detection to show the interpretability of SDG.

CCS CONCEPTS
• Mathematics of computing → Graph algorithms; • Computing methodologies → Neural networks; Learning latent representations;

KEYWORDS
Graph Neural Networks, Scalability, Interpretability

1 INTRODUCTION
Graphs are ubiquitous data structures nowadays for representing rich information regarding node interactions and connections. Combined with the powerful representation learning capabilities of deep neural network models, Graph Neural Networks (GNNs) [8, 11, 22] have achieved state-of-the-art performance in many high-impact applications, such as fraud detection [14], information retrieval [16], and recommender systems [23, 26].

2 PRELIMINARIES
In this section, we first introduce the notation used in this paper, and explain the basic theory behind our proposed model.
We use bold capital letters to denote matrices (e.g., $A$) and bold lower-case letters to denote column vectors (e.g., $a$). We follow the convention of Matlab for matrix indexing (e.g., $A(i,:)$ denotes the $i$-th row of $A$). Given an undirected and unweighted graph $G = (V, E)$ with nodes $V$ and edges $E$, $A \in \mathbb{R}^{n \times n}$ denotes the adjacency matrix with self-loops, $D \in \mathbb{R}^{n \times n}$ denotes the degree matrix with self-loops, $X \in \mathbb{R}^{n \times d}$ denotes the node feature matrix, and $Y \in \mathbb{R}^{n \times c}$ denotes the node label matrix, where $n$ is the number of nodes, $d$ is the dimension of node features, and $c$ is the dimension of node labels. Moreover, when the input graph $G$ changes with inserted and deleted edges $\Delta E$, we use $G'$ to denote the new graph. For the dimension consistency of matrices during the proposed dynamic propagation scheme, we consider the number of nodes to be fixed, i.e., an inserted (or deleted) node is regarded as a previous (or existing) dangling node [21].

In [25], authors have shown that the influence of a node $V_j$ on a node $V_i$ through a $k$-layer GNN model [8, 11] is proportional to the $k$-step random walk distribution on node $V_j$ starting from the seed node $V_i$. Also, when $k \to \infty$, the random walk distribution converges to the stationary distribution. That paves the way for improving the scalability of GNNs by replacing the traditional message-passing mechanism with the stationary distribution (i.e., personalized PageRank) to propagate information among nodes [2, 12], where the training time and the number of parameters are reduced for avoiding stacking layers in neural networks.

3 PROPOSED MODEL

In this section, we first introduce the overall framework of our proposed SDG graph neural network model, and then we illustrate each component of SDG in a systematic way.

![Figure 1: Structure of SDG Graph Neural Network.](image)

3.1 Overview of SDG

The proposed SDG graph neural network model has an input layer and an output layer as shown in Figure 1. To dynamically use the stationary distribution to propagate the neighborhood information among nodes instead of stacking layers, we propose the dynamic propagation scheme in our SDG graph neural network, which is expressed as follows.

$$Z = \text{softmax}(PH)$$  \hspace{1cm} (1)

where $P \in \mathbb{R}^{n \times n}$ is the dynamic propagation matrix, $H \in \mathbb{R}^{n \times c}$ is the hidden node feature matrix extracted from the input node features $X \in \mathbb{R}^{n \times d}$ through the model-agnostic neural network $f_0$, i.e., $H = f_0(X)$, and $Z \in \mathbb{R}^{n \times c}$ denotes the predicted label matrix.

Next we introduce how to form the dynamic propagation matrix $P$ for the changed graph structure in Subsection 3.2, and how to design the model-agnostic neural network $f_0$ to extract the hidden node feature matrix $H$ from the input node feature matrix $X$ in Subsection 3.3.

Finally, the loss function of SDG is defined as follows.

$$L_{\text{SDG}} = - \sum_{i}^{n} \sum_{j}^{c} Y(i, j) \ln Z(i, j)$$  \hspace{1cm} (2)

where $Y \in \mathbb{R}^{n \times c}$ and $Z \in \mathbb{R}^{n \times c}$ are the ground truth and model output respectively.

3.2 Dynamic Propagation Scheme

The proposed dynamic propagation scheme is realized by tracking the dynamic propagation matrix $P$, where each row $P(i,:) = \mathbf{0}$ is occupied by the stationary distribution of random walks starting from that node $V_i$. To be specific, $P(i,:) = \mathbf{0}$ can be fast tracked instead of solving from scratch when the graph topology changes from $A$ to $A'$. And the computation of each row is independent, which implies that the tracking process from $P$ to $P'$ can be parallelized. We first introduce the tracking process of $P'$, and then analyze its time complexity and error bound.

As each row $P(i,:)$ encodes the stationary distribution of random walks starting from node $V_i$, it can be expressed as follows.

$$P(i,:) = \alpha MP(i,:) + (1 - \alpha)r$$  \hspace{1cm} (3)

where $M = AD^{-1} \in \mathbb{R}^{n \times n}$ denotes the column-stochastic transition matrix, $\alpha \in [0, 1]$ denotes the teleportation probability, and $r \in \mathbb{R}^n$ denotes the personalized vector with $r(i) = 1$ and other entries equal to 0. For the clear notation, we omit the transpose notation $\top$ for $P(i,:)$ in the following part.

When the graph topology changes from $M$ to $M'$ with the updated edge $\Delta E$, the stationary distribution needs to be updated. To obtain $P'$, we need update each row of $P$. The core idea is to push out the previous probability distribution score from the changed part to the residual part of the graph [6, 27], and then add the updated distribution back to the previous distribution in order to finally obtain the new distribution. The tracking process can be described as follows.

$$P(i,:)_{\text{pushout}} = \alpha(M' - M)P(i,:)$$  \hspace{1cm} (4)

and

$$P(i,:) = P(i,:) + \sum_{k=0}^{\infty} (\alpha M')^k P(i,:)_{\text{pushout}}$$  \hspace{1cm} (5)

where $P(i,:)_{\text{pushout}}$ denotes the distribution score that needs to be pushed out on the residual graph due to the updated edges $\Delta E$, and $P(i,:) = \mathbf{0}$ denotes the tracked new distribution.

This push out process can be proved to converge to the exact stationary distribution of the new graph through sufficient cumulative power iterations [27, 28], i.e., $k \to \infty$. With this push out process, we can update each row of dynamic propagation matrix $P$ to further track $P'$. Next we analyze the time complexity and accuracy of approximately tracking the whole dynamic propagation matrix $P'$ with a threshold $\varepsilon$ as follows.

**Theorem 1.** Given the updated matrix $M'$, the threshold $\varepsilon$, and $q$ distributed machines, each machine at most costs $O(\frac{mn}{q} \log_2(\frac{1}{\varepsilon}))$ time to obtain the tracked dynamic propagation matrix $P'$ with each row bounded by $\frac{1}{1-\alpha}$ error compared with the stationary distribution.
in terms of $\ell_1$-norm, where $m$ is the number of non-zero entries of $M'$, and $n$ is the number of nodes in the graph.

Proof. With the threshold $\epsilon$, if $\|\alpha M'\|\leq \epsilon$ then we stop the iteration of Eq. 5. Since $M'$ is a column-stochastic matrix, then $\|M'\|_1 = 1$. Thus, the number of iterations $k$ satisfies $k \leq \log_{1/\alpha}(\|P(i,:)\|)/\epsilon$; and $P(i,:) = \alpha(M' - M)P(i,:)$, then $\|P(i,:)\|_1 \leq 2\alpha$. The computation of each row is independent, such that each distributed machine at most costs $O(m \log_{1/\alpha}(1/\epsilon))$ complexity. Also, denoting the early stop at the s-th iteration, then the remaining tracking error is denoted as $\sum_{k=s+1}^{\infty} \|\alpha M'\| P(i,:)\|_1$, which can be proved less than $\frac{\epsilon}{1-\alpha}$ just by replacing $P(i,:)\|_1$ with $\alpha(M' - M)P(i,:)$ and setting $k = \log_{1/\alpha}(\frac{1}{\epsilon})$.

Thus, tracking the whole dynamic propagation matrix $P'$ costs $O(m \log_{1/\alpha}(1/\epsilon))$ time complexity with the error bound $\frac{\epsilon}{1-\alpha}$ to avoid solving it from scratch by matrix inversion with $O(n^2)$ [12].

3.3 Model-Agnostic Neural Networks

The intuition of applying the model-agnostic neural network $f_0$ is to extract the qualified hidden node features $H$ from the input node features $X$. $f_0$ can take a variety of forms, such as CNNs [7]. Without loss of generality, we use a linear multilayer perceptron as $f_0$, to avoid increasing too much training complexity. Note that $f_0$ operates on each input node independently, which implies that extracting hidden node features could also be paralleled as follows.

$$H(i,:) = f_0(X(i,:))$$

where $X(i,:)$ denotes the input node features of node $V_i$, and $H(i,:)$ denotes the hidden node feature of node $V_i$ extracted by $f_0$.

Since the dynamic propagation $P$ can be pre-computed independently ahead of the training process of $f_0$, we separate the information propagation from the node feature extraction. It makes the scalability and interpretability of SDG neural network possible, when the input graph structure or the input node feature changes.

Scalability of SDG. When the structure of the input graph changes (i.e., from $A$ to $A'$) and/or the input node features change (i.e., from $X$ to $X'$), the dynamic propagation matrix $P'$ can be fast obtained in parallel as mentioned above. In this case, $f_0$ only needs to be fine-tuned to produce $H'$, starting from the previously well-trained parameters.

Interpretability of SDG. To investigate the influence of a certain node $V_i$ on the final prediction, SDG enables two types of methods, i.e., from the node feature view or from the graph structure view. First, we remove the hidden node features $H(i,:)=0$ and keep all remaining parts to see the change of the prediction results on other nodes. Second, we mask (i.e., delete) certain edges related to $V_i$ on the graph, fast track the new stationary distribution, and keep all remaining parts to see the change of the final predictions.

3.4 Optimization

We summarize the training process of SDG in Algorithm 1. In Steps 1-4, the initial inputs could be obtained from existing static algorithms like [2, 12]. Then in Steps 5-9, SDG incrementally tracks the dynamic propagation matrix and fine-tunes the model-agnostic neural network parameters if any change happens on the graph structure level and/or the node feature level.

4 EXPERIMENTS

In this section, we first introduce the data sets used for effectiveness comparison and efficiency analysis with state-of-the-art baseline algorithms in node classification tasks1. Then, to show the interpretability of our SDG model, we design the fake news detection case study by using the proposed SDG graph neural network model to classify news article word graphs, and to evaluate the influence of each word on the prediction result.

### Table 1: Dataset Statistics

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Classes</th>
<th>Nodes</th>
<th>Edges</th>
<th>Label Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Citeseer</td>
<td>6</td>
<td>2,110</td>
<td>3,668</td>
<td>0.036</td>
</tr>
<tr>
<td>Cora-ML</td>
<td>7</td>
<td>2,810</td>
<td>7,981</td>
<td>0.047</td>
</tr>
<tr>
<td>PubMed</td>
<td>3</td>
<td>19,717</td>
<td>44,324</td>
<td>0.003</td>
</tr>
</tbody>
</table>

4.1 Data Sets

We use three real-world citation graphs (i.e., Citeseer [20], Cora-ML [1,17], and PubMed [19]) to design experiments for the text classification problem. In the citation graph, each node represents a paper, and each edge represents the citation relationship between two papers. We leverage the largest connected component of each graph during the experiment, and each node feature is extracted by a bag-of-words representation of that paper’s abstract. The statistics of the three data sets are shown in Table 1.

4.2 Baseline Algorithms

PPNP and APPNP are two simplified graph neural network models [12] with state-of-the-art effective and efficient performance in many graph mining tasks [2,12]. Also, we include a variate of our SDG called SDG-S for the ablation study, which removes the dynamic propagation scheme, i.e., $k$ always equals to 0 in Eq. 5. With this variate, we can investigate the capability of the proposed dynamic propagation scheme in terms of improving effectiveness and efficiency.

1https://github.com/DongqiFu/SDG
Table 2: Effectiveness and Efficiency Comparison

<table>
<thead>
<tr>
<th>Methods</th>
<th>Citeseer</th>
<th>Cora-ML</th>
<th>PubMed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accuracy (%)</td>
<td>Time Consumption (s)</td>
<td>Accuracy (%)</td>
</tr>
<tr>
<td>PPNP</td>
<td>74.07±0.53</td>
<td>10.89±0.91</td>
<td>84.40±0.18</td>
</tr>
<tr>
<td>APPNP</td>
<td>73.93±0.30</td>
<td>22.80±1.69</td>
<td>84.63±0.34</td>
</tr>
<tr>
<td>SDG-S</td>
<td>74.10±0.30</td>
<td>6.65±0.69</td>
<td>84.60±0.28</td>
</tr>
<tr>
<td>SDG</td>
<td>74.17±0.39</td>
<td>7.11±0.93</td>
<td>84.87±0.68</td>
</tr>
</tbody>
</table>

4.3 Effectiveness and Efficiency Comparison

Due to the difference between static and dynamic graph neural network models, we mask 1% – 3% edges of the whole graph to form the graph $G$ and add back those edges to form the graph $G'$. After setting $\alpha = 0.9$ and making all algorithms converge at the same error threshold, we report the average node classification accuracy and time consumption on graph $G'$ of all baselines. Note that PPNP and APPNP are trained solely on graph $G'$, SDG and SDG-S are trained on graph $G$ and fine-tuned on graph $G'$. In Table 2, it can be observed that SDG and SDG-S could fast provide competitive performance for node classifications. To be specific, in PubMed, SDG achieves 84.70% accuracy that is 0.79% higher than the third best (PPNP). An intuitive explanation is that the tracked stationary distribution of SDG is more suitable for the parameters of the model-agnostic neural network to classify nodes. Moreover, SDG-S also provides the acceptable accuracy in a fast manner for avoiding the tracking process only with the outdated stationary distribution. A possible explanation is that the topology update between two graphs $G$ and $G'$ is relatively small as compared to the whole graph, and the model-agnostic neural network bridges the insufficient tracking with its adequate representative abilities. To verify this explanation, we design the following parameter sensitivity analysis.

4.4 Parameter Sensitivity

To verify the above mentioned explanation, we change the value of two important parameters, i.e., the teleportation probability $\alpha$ and the number of tracking iterations $k$, and to see how the performance changes accordingly. In Figure 2, as the number of tracking iterations increases, the classification accuracy increases but only to a small extent. This observation suggests that the model-agnostic neural network bridges the gap between approximated tracked stationary distribution and the real stationary distribution, which means the classification accuracy is not very sensitive to $k$ when the graph structure does not change too much. Therefore, the proposed SDG model can be readily applied to interpret the prediction result, i.e., when only a small portion of the input graph is masked, a small $k$ can also provide comparable performance.

4.5 Case Study

Disinformation dissemination has a huge negative impact on our society, and many factors may hinder the identification of disinformation. For example, messages on the social network are usually short and fast propagated, which requires that the predictive model should have efficient training process and adaptable structures for the evolving environment. Hence, we design the fake news case study with the proposed SDG model and a fake news data set [13].
REFERENCES


